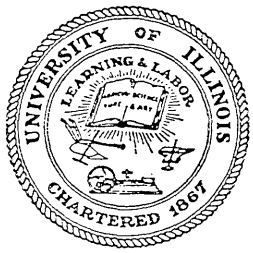
UILU-ENG-78-2020

CIVIL ENGINEERING STUDIES
STRUCTURAL RESEARCH SERIES NO. 454



NUMERICAL AND SOFTWARE REQUIREMENTS FOR GENERAL NONLINEAR FINITE ELEMENT ANALYSIS

By
R. H. DODDS, JR.
L. A. LOPEZ
D. A. PECKNOLD

A Technical Report of
Research Issued by
THE OFFICE OF NAVAL RESEARCH
DEPARTMENT OF THE NAVY
Contract No. N00014-75-C-0164
Project No. NR 064-183

UNIVERSITY OF ILLINOIS
at URBANA-CHAMPAIGN
URBANA, ILLINOIS
SEPTEMBER 1978

NUMERICAL AND SOFTWARE REQUIREMENTS

FOR GENERAL NONLINEAR FINITE ELEMENT ANALYSIS

by

R. H. DODDS, JR.
L. A. LOPEZ
D. A. PECKNOLD

UNIVERSITY OF ILLINOIS
AT URBANA-CHAMPAIGN
URBANA, ILLINOIS
SEPTEMBER, 1978

ACKNOWLEDGMENT

This report is based on the dissertation of Robert H. Dodds, Jr. submitted to the Graduate College, University of Illinois at Urbana-Champaign, for the degree of Doctor of Philosophy in Civil Engineering. The study was conducted under the direction of Dr. L. A. Lopez.

Financial support provided by a University of Illinois Fellowship and Research Assistantships is gratefully acknowledged.

Dr. Dodds wishes to express his sincere appreciation to Professors L. A. Lopez, D. A. Pecknold, and W. C. Schnobrich for their interest and guidance throughout this study.

Acknowledgments are also due to the Civil Engineering Systems Laboratory (CESL) of the University of Illinois for unlimited use of the Burroughs B-6700 computer. This project could not have been completed without the use of this reliable resource.

Reproduction of this report was made possible by the Office of Naval Research, Contract No. N00014-75-C-0164.

TABLE OF CONTENTS

CHAPTER Page

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

1.1 General

Analytical solution of most practical nonlinear structures is currently not possible. The complex nonlinear partial differential equations of equilibrium arising from irregular geometry, large displacements, and inelastic material behavior are intractable with current analytical techniques. In many instances, the governing equations cannot even be explicitly written in closed form.

Consequently, nonlinear analysis of practical structures is possible only with approximate numerical techniques. Finite differences and lumped parameter, or analog, models were once popular methods to formulate and solve the differential equations. Such methods are well suited for two-dimensional structures of regular geometry. However, they are cumbersome for irregular geometries, three dimensional problems, and transition regions between low and high stress gradients.

The finite element method (FEM) developed during the past two decades provides an alternative modeling technique for structural analysis. A structure is first idealized by an assemblage of discrete pieces, or elements. Within each element, simple expressions approximate the response as functions of unknown quantities specified at common points (nodes) between elements. Then, instead of numerically expressing the partial differential equations of equilibrium, variational principles are employed to generate an approximate set of equilibrium equations directly in terms of the field variables, usually nodal displacements. The FEM

has proven highly successful in linear analysis because of its ability to model complex geometries and variable material properties, and to combine various types of structural elements. The FEM is currently proving to be even more powerful for nonlinear analysis and is the focal point of this work.

Development of the FEM coincided with, and was prompted by, concurrent advances in digital computer technology. Without modern computers, the FEM would be relatively useless for practical analysis. The enormous number of computations involved in setting up and solving the governing equations is particularly well suited for computer application. However, finite element computer software has received surprisingly little formal attention as a research topic. Most researchers are concerned with theoretical formulations and practical applications of the FEM. Only recently has the importance of "engineered" finite element software been realized by the profession. The true beauty and power of the FEM is not realized until one has analyzed an extremely complex structure with a well designed user-oriented finite element software system.

This work has attempted to survey the state-of-the-art in nonlinear finite element technology and to design and implement a comprehensive, user-oriented software system that brings these techniques to practicing engineers, researchers, and future system developers.

1.2 Current Nonlinear Finite Element Systems

Currently available finite element systems can be separated into three distinct categories according to internal structure, ease of use, and overall generality. Complete reviews of these and many special-purpose

programs for linear and nonlinear analysis can be found in Ref. [55].

A) MARC [41], ANSYS [68], and ADINA -- Both MARC and ADINA (a revised NONSAP [7]) resulted from university research projects on nonlinear finite element and material model formulations. ANSYS was developed in private industry for linear analysis and later extended to include a nonlinear capability. MARC and ANSYS were developed during the late 1960's whereas ADINA is relatively recent (1975). Free versions of these systems are not supported and generally are of little practical value. Proprietary versions have been extensively upgraded and are now in widespread use.

Each program has geometric and material nonlinear capability with solid mechanics as the primary application area. MARC has the more extensive nonlinear material modeling capability. Time history integration and modal analysis features enable nonlinear dynamic analysis. Solution procedures are based on incremental or marching techniques that trace the approximate load-displacement path. Recent versions of these programs support the general Newton-Raphson method that corrects for errors introduced during the incremental solution.

The popularity of these programs is due to their being the first to offer new finite elements, nonlinear material models, and equation solvers in a usable form. However, they have some undesirable characteristics. For example, input is generally through fixed-format lines of data in the batch mode (the most recent versions are beginning to support a question-and-answer interactive dialogue type of input). These programs are generally not complete. Frequently, users must develop extensive pre and post processing packages that reduce engineering time necessary to specify the structural model and interpret the results.

The internal program and data structures found in these systems are typical of software generated by university research projects. The simple vector and matrix data structure concepts supported directly by FORTRAN are utilized. No integrated data management or virtual memory techniques are employed. The advantages of this approach are short development time and low development cost.

There are significant disadvantages associated with this type of finite element software. From the user's point of view, these programs appear as a hodgepodge of unrelated packages. Often the user must execute a series of programs to complete a single analysis. Addition of new features can require extensive re-writes of the system. Extensions to add new types of finite elements and material models are especially difficult to implement, since no formal communication interfaces exist between program processing routines and element-material model routines. Similarly, element and material model routines are not logically separate -- element routines invoke material model routines directly thus generating hidden lower level interfaces. Both of the above factors greatly complicate the process of adding new elements and models. These disadvantages are of no serious consequence for the casual user, but researchers interested in augmenting the system must have knowledge of the internal organization, existing elements, and material models in order to install new elements and models. In addition, developers must program many tasks such as input, output, and memory allocation, that are not directly related to the element or material model. These tasks are common to all elements and material models and should be performed by the system itself.

B) NASTRAN [38], ASKA [5,66] -- These systems are considerably more complex in terms of internal organization and data structure than those of A) and were developed at great expense (millions of dollars) through government funding. They were originally designed for static and dynamic analysis of very large linear structures. Neither system has a nonlinear capability comparable to the research oriented programs of A). ASKA is currently undergoing an extensive re-write at ISD; at this time no information is available on planned nonlinear capabilities. Proprietary versions of the NASTRAN system are the only desirable ones available today. These have updated element libraries and are considerably more efficient than the public domain versions; few nonlinear extensions have been implemented. Both the ASKA and NASTRAN systems still receive considerable financial support from government and private industry

The popularity of NASTRAN and ASKA stems from their being the first to solve very large practical structures. They are still not user-oriented. Input is generally fixed-format, although ASKA now does have limited free-form input. Numerous pre and post processing packages are available, especially for the NASTRAN program.

Developers of new elements and material models face the same difficulties as with the programs of A). System-element interfaces are not formalized thereby requiring a knowledge of the system organization to install new elements and nonlinear material models.

NASTRAN and ASKA are based on a concept that evolved during the mid 1960's known as "programming systems". With this scheme, a number of processing modules are designed to perform specific operations on data blocks (NASTRAN) or hypermatrix data structures (ASKA). The finite element

system then consists of appropriate sequences of calls to the data block and hypermatrix processors. NASTRAN and ASKA have pre-programmed sequences or "formats" for standard analysis procedures; i.e., static linear analysis, modal analysis, time history integration, etc.

Processing modules request data through a separate data management system. NASTRAN employs DMAP-GINO, a filing system that operates in a simple matrix extraction mode. Data management functions in ASKA are performed with DRS (Data Retrieval System) that operates on hypermatrix data structures stored physically as orthogonal lists.

C) STRUDL [31], FINITE [34,35] -- These two systems are based upon the concept of integrated engineering software -- a significantly different approach than used in systems of B). Both use sophisticated supervisory systems that aid engineers in the development of complex, user-oriented software for all types of engineering problems, not just finite element analysis. Data structures considerably more general than hypermatrices are supported thereby eliminating many of the difficulties encountered in systems of B).

STRUDL operates under the ICES supervisor [61], while FINITE operates under the POLO II supervisor [32,33,37]. The two supervisors perform essentially the same tasks; however, the philosophy of implementation is radically different.

ICES provides software development support through ICETRAN, a language similar to FORTRAN but with extended data structures and memory management. ICETRAN is converted to standard FORTRAN by the ICES precompiler which inserts numerous statements referring to ICES run time

support routines. These routines are merged with FORTRAN emitted by the precompiler to form a complete program. The ICES run time support routines were written in machine language thereby making the supervisor highly machine dependent.

POLO, developed after ICES was operational, provides more comprehensive data management capabilities and operates in a compiled interpretive mode. The run time support routines are written in FORTRAN thus making POLO considerably more machine independent than ICES. Additional details of the POLO system relative to nonlinear finite element software are given in Chapter 5.

Emphasis during the development of STRUDL was on ease of defining the structural model as well as the computational aspects of analysis. However, it was originally designed only for frame analysis; a finite element capability was added afterwards. Some versions of STRUDL offer a limited geometric nonlinear capability for buckling analysis of frames. Nonlinear material behavior is not supported.

The FINITE system was designed to have the same ease of use as STRUDL but with greatly extended structural modeling capabilities through multi-level substructuring and static condensation. FINITE is a true finite element system; the frame analysis features so popular in STRUDL are incorporated in the same manner as any other type of finite element.

A distinct feature of the FINITE system involves a formal element "definition" procedure. The system-element communication interface is standard for all elements. Knowledge of system organization is not required to implement new elements. In addition, FINITE performs many

features common to all elements that must be programmed by developers in the systems already discussed. As part of this study, the formal "definition" procedure was extended to include both nonlinear finite elements and material models.

1.3 Objectives and Scope

The objectives of the research reported herein are threefold:

1) To summarize the equations of nonlinear continuum mechanics in matrix form suitable for application to finite element analysis and to determine the more appropriate formulation for incorporating large displacement effects;
2) To design a user-oriented, general static nonlinear finite element system that includes such features as multi-level substructuring and condensation. The internal organization, analytical features, and user-system interfaces proposed here should serve as a model for software to satisfy the needs of practicing engineers and researchers. Particular emphasis in this work is placed on interaction between the system and finite element and material model researchers.
3) To implement the above within the POLO-FINITE structural mechanics system to show that the proposed concepts are feasible and efficient for solution of nonlinear finite element problems.

Correspondingly, this report is divided into chapters that present the techniques used to achieve the objectives.

Currently available numerical formulations for nonlinear finite element analysis are reviewed in Chapters 2 and 3 to establish those

procedures applicable to a wide class of problems. Both geometric and material nonlinearities are considered. Finite strain behavior is incorporated within the Lagrangian formulation adopted.

Chapter 4 examines the various aspects of finite element software that directly affect users. This includes user-system communication, structural modeling through substructuring and condensation, and automatic analysis restart.

Chapter 5 is an overview of the nonlinear POLO-FINITE system as designed and implemented during this work. Particular topics addressed include subsystem and data base structure, nonlinear material modeling for the analyst and material behavior researcher, and processors for directing the nonlinear analysis.

The solutions to several example problems are presented in Chapter 6 to illustrate the applicability of FINITE's nonlinear capabilities.

Chapter 7 briefly summarizes this research effort.

CHAPTER 2

NONLINEAR STRUCTURAL MECHANICS

2.1 General

All rational approaches to the development of nonlinear finite element formulations rely upon the basic principles of nonlinear continuum mechanics. This chapter presents a concise summary of the salient features of nonlinear continuum mechanics in a form suitable for direct application to the finite element method.

The components of nonlinear continuum theory necessary to develop a finite element formulation are:

1) Strain-displacement relations;
2) Stress definitions;
3) Constitutive equations relating stress to strain;
4) Virtual work principles;

Equations describing each of the above components are given in terms of a general three dimensional body with respect to rectangular coordinates. Specialization of these general equations result in explicit formulations for oriented bodies such as rods, beams, plates, and shells.

No attempt has been made here to derive all the equations from first principles. Rather the results from several classical treatises are summarized in matrix notation familiar to most structural engineers. For detailed derivations, the reader is referred to the works of Murnaghan [45], Novozhilov [53], and Oden [54].

2.2 Coordinate Systems

There are essentially two unique ways to describe the deformation of a body. In the first, all quantities of interest are expressed as functions of the rectangular coordinates of the body in its initial or undeformed state. Such a formulation is termed Lagrangian or Total Lagrangian in the literature. Strain components of linear elasticity, given in terms of displacement derivatives with respect to the initial coordinates, are derived from a Lagrangian approach.

In the second approach, termed Eulerian, convected coordinates, and Updated Lagrangian, rectangular coordinates are continually updated to reflect the changing configuration of the body. Subsequent changes of displacement, strain, and stress are expressed as functions of the instantaneous coordinates. Linear elasticity makes implicit use of the Eulerian system to derive the equilibrium equations by considering an infinitesimal volume isolated from the deformed body.

If only infinitesimal displacement derivatives are present, the two approaches are essentially identical. However, if large displacement gradients occur, resulting in excessive rotations and/or deformation, governing equations derived from the two methods are different. The choice of formulations is a matter of convenience. The Lagrangian approach is natural for expressing the deformed geometry and strain of a body while equilibrium equations are simplest in the Eulerian formulation. Geometric transformations relating the initial and deformed configurations of a body permit the use of both formulations in a finite element analysis. These transformations are derived in the following sections.

Consider an arbitrary three dimensional body in an initial configuration (taken as undeformed for simplicity) as shown in Fig. 2.2.1 All

points P(x,y,z) are located relative to a rectangular coordinate system {X} with unit vectors {i} directed along the axes. As a result of an applied loading, each point P undergoes a displacement {U(x,y,z)} with projections onto the unit vectors {i} denoted by [u,v,w]. The new coordinates of points in the body, {X'} are given by

$$\{X'\} = \{X\} + \{U\} \tag{2.2.1}$$

where the terms of {X'} are functions of the initial coordinates. The Jacobian matrix of the functions {X'} relates differentials in the {X} and {X'} systems by

$$\{dX'\} = [J]\ \{dX\} \tag{2.2.2}$$

where,

$$\{dX\}^T = [dx,\ dy,\ dz] \tag{2.2.3}$$

$$\{dX'\}^T = [dx',\ dy',\ dz'] \tag{2.2.4}$$

and,

$$[J] = \frac{\partial(X',Y',Z')}{\partial(X,Y,Z)} = \begin{vmatrix} X'_x & X'_y & X'_z \\ Y'_x & Y'_y & Y'_z \\ Z'_x & Z'_y & Z'_z \end{vmatrix} \tag{2.2.5}$$

The Jacobian matrix must possess a positive determinant for all points P(x,y,z). A positive determinant assures a unique relationship exists between the initial and deformed configuration, i.e., two points in the initial configuration must occupy two unique positions in the deformed

configuration. Furthermore, the [J] matrix defines a linear mapping of lines, surfaces, and volumes in the infinitesimal neighborhood of each point P. Straight lines in the initial configuration map into straight lines in the deformed configuration. Similarly, planes map into planes and parallelism of straight lines and planes is preserved.

During deformation, coordinate lines originally parallel to the {X} axes are distorted into a nonorthogonal curvilinear coordinate system termed $\{\tilde{X}\}$ as shown in Fig. 2.2.2 for a two dimensional body. The Jacobian matrix contains the information describing the distortion. Consider a point $P_0(x,y)$ and a neighboring point $P_1(x + dx, y + dy)$ that enclose an infinitesimal rectangle of area dxdy. After displacements have occurred, the rectangle is distorted into a parallelogram with new corner coordinates as shown in the figure. In particular, sides $\bar{i}\,dx$ and $\bar{j}\,dy$ are distorted into vectors $\tilde{i}$ and $\tilde{j}$. From simple geometry, the projections of $\tilde{i}$ and $\tilde{j}$ onto {i} are given by the columns of [J]. The area of the parallelogram is given by $\tilde{i} \times \tilde{j}$ or simply det[J]dxdy. The quantities {dX'} are then the projections of the line joining the points P_0 and P_1 in the deformed configuration on unit vectors {i}. Before deformation the projections were {dX}.

A differential tetrahedron in the initial configuration with centroid at point P_0 is shown in Fig. 2.2.3. The diagonal face has area dS with an outward unit normal {n}. Projections of dS onto the coordinate planes are

$$\{dA\} = dS\ \{n\} \tag{2.2.6}$$

where for example dA_x is the projection of dS onto the YZ plane. After deformation the area dS is distorted into dS' with a unit outward normal

{n'}. The projections of dS' onto the coordinate planes are denoted {dA'}. By considering areas initially perpendicular to the coordinate axes {X} and following the same procedure as above for the two dimensional case, Murnaghan [45] showed the relationship between {dA'} and {dA} can be written as

$$\{dA'\} = |J|\ [J^T]^{-1}\ \{dA\} \tag{2.2.7}$$

where the magnitude of area change is related to both $|J|$ and $[J^T]^{-1}$ with the change in orientation of dS provided by $[J^T]^{-1}$. Similarly, an area in the deformed configuration with projections {dA'} would have projections {dA} given by

$$\{dA\} = |J|^{-1}\ [J]\ \{dA'\} \tag{2.2.8}$$

A differential rectangular volume, dV, in the initial configuration becomes a parallelopiped with volume dV' in the deformed configuration. The ratio of deformed and undeformed volumes is given by

$$dV' = |J|\ dV \tag{2.2.9}$$

This follows from a direct application of the triple scalar product to compute the volume of a parallelopiped using the columns of [J] as vectors tangent to each edge.

2.3 Strain-Displacement Relations

A major source of nonlinear behavior arises when finite rather than infinitesimal, displacement gradients and rotations occur in a body. In this section the relations between strain and displacement for arbitrarily

large displacement gradients are presented in a form suitable for finite element analysis.

2.3.1 Strain Definitions

The definition of "engineering" strain was first proposed by Cauchy (1827) and is of fundamental importance in linear elasticity. Cauchy defined strain in terms of the initial and final lengths of a fiber in the body as

$$\varepsilon_c = (L_f - L_i)/L_i \tag{2.3.1}$$

where ε_c denotes Cauchy strain. L_i and L_f are the initial and final lengths of a fiber.

In the theory of finite deformation, two definitions of direct strain are commonly employed. The first, due to Green (1839), expresses deformation as a function of the initial configuration (Lagrangian formulation). Green's direct strain is given by

$$\varepsilon_g = (L_f^2 - L_i^2)/2L_i^2 \tag{2.3.2}$$

The factor 1/2 makes Green strain equal to Cauchy strain for small length changes, i.e., $L_f L_i$ is approximated by L_i^2. The second definition of finite direct strain, due to Almansi (1911), expresses strain as a function of the deformed configuration (Eulerian formulation). Almansi's direct strain is given by

$$\varepsilon_a = (L_f^2 - L_i^2)/2L_f^2 \tag{2.3.3}$$

Again the factor 1/2 is required as for Green's strain.

2.3.2 Green Strain Components

Components of direct Green strain for general three dimensional bodies are obtained from the Jacobian matrix of Eq. 2.2.5. Consider two points a differential distance ds apart on a space curve C before deformation as shown in Fig. 2.3.1. The projections of ds on the unit vectors {i} are simply {dX}. The magnitude of ds is the positive square root of $\{dX\}^T\{dX\}$. As a result of deformation the curve C is distorted into a new curve C'. The distance between the two points after deformation is ds' with projections {dX'} on unit vectors {i}. Defining ds' as the positive square root of $\{dX'\}^T\{dX'\}$, the following relation is written

$$(ds')^2 - (ds)^2 = \{dX'\}^T \{dX'\} - \{dX\}^T \{dX\} \qquad (2.3.4)$$

The quantity $(ds')^2 - (ds)^2$ defines the deformation in the infinitesimal neighborhood of the point P(x,y,z). Writing {dX'} in terms of {dX} using [J] yields

$$(ds')^2 - (ds)^2 = \{dX\}^T [J^TJ - I] \{dX\} \qquad (2.3.5)$$

If the distance between two points before and after deformation remains unaltered, it follows that $[J^TJ] = [I]$ and that [J] is a simple rotation matrix. For $[J^TJ] \neq [I]$, the difference corresponds to that part of the displacements not causing rigid body motion and is therefore a measure of the deformation at a point. In view of the above, the components of Green strain are thus given by

$$[\varepsilon_g] = [J^TJ - I]\ 1/2 \qquad (2.3.6)$$

The 3 x 3 $[\varepsilon_g]$ matrix contains all terms necessary to describe the finite

deformation of a body in the Lagrangian formulation. From Eq. 2.3.6 the strain components are seen to be symmetric and are usually written in vector form $\{\varepsilon_g\}$ for finite element applications.

Alternatively, $[\varepsilon_g]$ can be written in terms of displacement derivatives by noting that

$$[J] = [j] + [I] \tag{2.3.7}$$

where

$$[j] = \begin{bmatrix} U_x & U_y & U_z \\ V_x & V_y & V_z \\ W_x & W_y & W_z \end{bmatrix} \tag{2.3.8}$$

Substituting into Eq. 2.3.6

$$[\varepsilon_g] = [j + j^T + j^T j] \; 1/2 \tag{2.3.9}$$

Performing the operations indicated above, the components of Green strain in vector form $\{\varepsilon_g\}$ are

$$\begin{aligned}
\varepsilon_x &= u_x + \frac{1}{2}[u_x^2 + v_x^2 + w_x^2] \\
\varepsilon_y &= v_y + \frac{1}{2}[u_y^2 + v_y^2 + w_y^2] \\
\varepsilon_z &= w_z + \frac{1}{2}[u_z^2 + v_z^2 + w_z^2] \\
\varepsilon_{xy} &= u_y + v_x + [u_x u_y + v_x v_y + w_x w_y] \\
\varepsilon_{xz} &= u_z + w_x + [u_z u_x + v_z v_x + w_z w_x] \\
\varepsilon_{yz} &= v_z + w_y + [u_y u_z + v_y v_z + w_y w_z]
\end{aligned} \tag{2.3.10}$$

where subscripts imply differentiation. Clearly, if products of derivatives are negligible, the components of Green strain simplify to those of linear elasticity.

2.3.3 Almansi Strain Components

Strains for the Eulerian formulation defined in terms of the instantaneous coordinates {X'} are those due to Almansi. The inverse of Eq. 2.2.2 provides {dX} in terms of {dX'}

$$\{dX\} = [J]^{-1} \{dX'\} \tag{2.3.11}$$

Thus, a line segment in the deformed body with projections {dX'} has projections {dX} given by the above if the deformations are reversed. The inverted Jacobian is given by

$$[\bar{J}] = [J]^{-1} = \begin{vmatrix} X_{x'} & X_{y'} & X_{z'} \\ Y_{x'} & Y_{y'} & Y_{z'} \\ Z_{x'} & Z_{y'} & Z_{z'} \end{vmatrix} \tag{2.3.12}$$

or in terms of displacement gradients,

$$[\bar{J}] = [I] - [\bar{j}] \tag{2.3.13}$$

where,

$$[\bar{j}] = \begin{vmatrix} U_{x'} & U_{y'} & U_{z'} \\ V_{x'} & V_{y'} & V_{z'} \\ W_{x'} & W_{y'} & W_{z'} \end{vmatrix} \tag{2.3.14}$$

To compute $[\bar{j}]$, the displacements $\{U\}$ are added to the coordinates $\{X\}$, then numerical evaluation of $[j]$ and $[\bar{j}]$ are identical.

Expressing the squared change in length of a line segment as a function of the $\{X'\}$ coordinates, Eq. 2.3.4 is rewritten as

$$(ds')^2 - (ds)^2 = \{dX'\}^T [I - \bar{J}^T\bar{J}] \{dX'\} \tag{2.3.15}$$

from which the components of Almansi direct strain are

$$[\varepsilon_a] = [I - \bar{J}^T\bar{J}] \; 1/2 \tag{2.3.16}$$

or in terms of displacement gradients

$$[\varepsilon_a] = [\bar{j} + \bar{j}^T - \bar{j}^T\bar{j}] \; 1/2 \tag{2.3.17}$$

Almansi strain components in vector form $\{\varepsilon_a\}$ are

$$\begin{aligned}
\varepsilon_{x'} &= u_{x'} - \tfrac{1}{2}[u_{x'}^2 + v_{x'}^2 + w_{x'}^2] \\
\varepsilon_{y'} &= v_{y'} - \tfrac{1}{2}[u_{y'}^2 + v_{y'}^2 + w_{y'}^2] \\
\varepsilon_{z'} &= w_{z'} - \tfrac{1}{2}[u_{z'}^2 + v_{z'}^2 + w_{z'}^2] \\
\varepsilon_{x'y'} &= u_{y'} + v_{x'} - [u_{x'}u_{y'} + v_{x'}v_{y'} + w_{x'}w_{y'}] \\
\varepsilon_{x'z'} &= u_{w'} + w_{x'} - [u_{x'}u_{z'} + v_{x'}v_{z'} + w_{x'}w_{z'}] \\
\varepsilon_{y'z'} &= v_{z'} + w_{y'} - [u_{y'}u_{z'} + v_{y'}v_{z'} + w_{y'}w_{z'}]
\end{aligned} \tag{2.3.18}$$

If the products of differentials are negligible and the displacements are infinitesimal, the above strain components simplify to the Cauchy strains.

2.3.4 Strain Increments

The principle of virtual work for large deformations derived in a subsequent section requires an expression relating strain increments to displacement increments. Increments of Green strain are derived below; Almansi strain increments are determined with an identical procedure.

Consider the components of Green strain defined in Eq. 2.3.10. For a variation of the displacements $\{\delta U\}$, the variation of $\delta\varepsilon_x$ is given by

$$\delta\varepsilon_x = \delta(u_x) + \frac{1}{2}\left[\delta(u_x)^2 + \delta(v_x)^2 + \delta(w_x)^2\right] \qquad (2.3.19)$$

which can also be written

$$\delta\varepsilon_x = (1 + u_x)\ (\delta u_x) + v_x\ (\delta v_x) + w_x(\delta w_x) \qquad (2.3.20)$$

The coefficients of the variational terms comprise the first row of the Jacobian matrix of Eq. 2.2.5. For infinitesimal displacements the derivatives of deformed coordinates $\{X'\}$ with respect to undeformed coordinates $\{X\}$ result in $[J] = [I]$ with the above variation in ε_x simplifying to the variation of Cauchy strain. Proceeding as above for each Green strain component, the resulting variation of the Green strain vector $\{\delta\varepsilon_g\}$ is written for the two dimensional case as

$$\{\delta\varepsilon_g\} = \begin{Bmatrix} \delta\varepsilon_x \\ \delta\varepsilon_y \\ \delta\varepsilon_{xy} \end{Bmatrix} = [B]\ \{\delta U\} \qquad (2.3.21)$$

where $[B]$ is a 3 x 2 differential operator defined by

$$[B] = \begin{bmatrix} X'_x \frac{\partial}{\partial x} & Y'_x \frac{\partial}{\partial x} \\ X'_y \frac{\partial}{\partial y} & Y'_y \frac{\partial}{\partial y} \\ X'_y \frac{\partial}{\partial x} + X'_x \frac{\partial}{\partial y} & Y'_y \frac{\partial}{\partial x} + Y'_x \frac{\partial}{\partial y} \end{bmatrix} \tag{2.3.22}$$

Axisymmetric and three dimensional expressions for the [B] matrix are derived in a similar manner.

2.4 Stress

In general, stress provides a measure of loading intensity on a specified area of a body. Relationships among stress components in linear theory are derived assuming negligible changes in geometry due to displacements. In such cases, stress can be adequately represented as functions of the initial coordinates {X}. However, when displacements cause large rotations and/or distortion of the body, several definitions of stress arise quite naturally as functions of either the initial or deformed coordinates. Care must be taken to assure that the proper stress definition is used when deriving the equilibrium equations. This section examines the three most common definitions of stress proposed in finite deformation theory, namely, Cauchy or Eulerian stress, Lagrangian or 1st Piola-Kirchoff stress, and the 2nd Piola-Kirchoff stresses. Equilibrium equations for the body are derived consistent with each definition of stress.

2.4.1 Cauchy Stress

Consider a differential volume of a body that after deformation is a simple tetrahedron as shown in Fig. 2.4.1. The diagonal face has area dS' with unit outward normal {n'}. Projections of the vector area dS'{n'} on the coordinate planes are {dA'} as shown in the figure. An infinitesimal force, dT', with projections {dT'} along the unit vectors {i} acts over the area dS'. This force arises as the result of applied loads along a bounding surface or through interaction with adjacent elements of the body. The 3 x 3 matrix of stress components $[\sigma']$ acting on the projected areas {dA'} are shown in the figure. Equilibrium of the tetrahedron requires that stress components be related to the force by

$$\{dT'\} = [\sigma'] \{n'\} dS' \tag{2.4.1}$$

or,

$$\{dT'\} = [\sigma'] \{dA'\} \tag{2.4.2}$$

Elements of $[\sigma']$ are termed Eulerian, Cauchy, or true stresses since they reflect actual forces acting on the deformed areas.

Equilibrium conditions are then determined by equilibrating the resultant body force acting on any arbitrary volume, v', with the resultant of tractions applied over the boundary, S' of the arbitrary volume. Integrating Eq. 2.4.1 yields

$$\int_{V'} \{F'\} dV' = \int_{S'} [\sigma']\{n'\}dS' \tag{2.4.3}$$

where integration is with respect to the {X'} coordinates and extends over the deformed volume and surface area. The vector {F'} contains components of applied body force per unit deformed volume. Application of the divergence theorem to the right side of Eq. 2.4.3 converts the

surface integral to a volume integral with the resulting equilibrium conditions given as

$$\int_{V'} [\{F'\} + \mathrm{div}_{X'} [\sigma']] \, dV' = \{0\} \tag{2.4.4}$$

The above can hold for any arbitrary portion of the body only if the integrand vanishes everywhere in the body, resulting in the equilibrium equations in terms of stress components

$$\mathrm{div}_{X'} [\sigma'] + \{F'\} = \{0\} \tag{2.4.5}$$

Similarly, equating the total moments of body forces and surface forces about the origin shows that $\sigma'_{ij} = \sigma'_{ji}$. At first these equilibrium equations appear identical to those derived in linear elasticity. The difference is that the Eulerian stress components are functions of the coordinates $\{X'\}$ not $\{X\}$ and that differentiation is with respect to $\{X'\}$. However, derivatives with respect to $\{X'\}$ cannot be obtained until displacements are known. This difficulty is overcome by transforming the equilibrium equations into functions of the $\{X\}$ coordinates as shown in the following sections.

2.4.2 Lagrange and Piola-Kirchoff Stresses

The limits on integrals of Eq. 2.4.3 expressing equilibrium conditions extend over the deformed volume and surface area of the body. Limits on the integrals are transformed by writing

$$\int_V \{F'\}(dV'/dV) \, dV = \int_S [\sigma']\{n'\}(dS'/dS) \, dS \tag{2.4.6}$$

such that the integrals are now evaluated over the initial configuration of the body. Substituting for the components of $\{dA'\}$ in terms of $\{dA\}$ from Eq. 2.2.7 and for dV' from Eq. 2.2.9, the above equation can be

written

$$\int_V \{F'\}|J|\, dV = \int_S [\sigma']\ |J|\ [J^T]^{-1}\{n\}\, dS \qquad (2.4.7)$$

where the terms of the 3 x 3 matrix defined as

$$[\sigma_L] = [\sigma']\ |J|\ [J^T]^{-1} \qquad (2.4.8)$$

are called the Lagrange or 1st Piola-Kirchoff stress components. Lagrange stress components act in directions corresponding to unit vectors {i} over the initial area dS and equilibrate the same differential force {dT'} as do Eulerian stresses acting on the deformed area dS'. The same result is obtained by first transforming from dS' to dS in Eq. 2.4.1 then expressing the equilibrium condition.

The Jacobian matrix [J] is in general not symmetric thus rendering the components of Lagrange stress nonsymmetric as given by the transformation in Eq. 2.4.8. This proves to be inconvenient as both Green and Almansi strain components are symmetric thus necessitating a nonsymmetric constitutive relation even for isotropic materials.

An alternative definition of stress is derived by noting that Lagrange stresses acting on undeformed areas equilibrate the same force {dT'} acting on the deformed area. However, a force, {dT}, acting on the undeformed area can be defined by

$$\{dT\} = [J]^{-1}\{dT'\} \qquad (2.4.9)$$

This expression states that a force vector with components {dT} acting on the body before deformation has components {dT'} after deformation which is analogous to the changing direction of a vector connecting two points before and after deformation. Substituting for {dT'} into Eq. 2.4.1

and changing from {dA'} to {dA} yields

$$\{dT\} = [J]^{-1}\ [\sigma']\ |J|\ [J^T]^{-1}\ \{dA\} \tag{2.4.10}$$

from which the 3 x 3 matrix $[\sigma]$ is defined as

$$[\sigma] = [J]^{-1}\ [\sigma']\ |J|\ [J^T]^{-1} \tag{2.4.11}$$

contains components of the 2nd Piola-Kirchoff stress. This is a symmetric transformation; consequently components of 2nd Piola-Kirchoff stress are symmetric. These components are not true "stresses" in the same sense as Eulerian and Lagrange stresses since they equilibrate a transformed force vector. However, the symmetry of these stresses makes them most convenient. Their definition is not arbitrary as will be seen in the virtual work equations where they arise quite naturally. The Lagrange and 2nd Piola-Kirchoff stress components are related by

$$[\sigma_L] = [J]\ [\sigma] \tag{2.4.12}$$

Substituting the above relation into Eq. 2.4.7 yields the equilibrium equations in terms of 2nd Piola-Kirchoff stress

$$\int_V \{F'\}\ |J|\ dV = \int_S [J]\ [\sigma]\ \{n\}\ dS \tag{2.4.13}$$

with $\{F\} = \{F'\}|J|$ the body force per unit of initial volume. Eq. 2.4.13 is an exact statement of equilibrium. Applying the divergence theorem to the right side yields the equilibrium equations in terms of 2nd Piola-Kirchoff stresses

$$\mathrm{div}_X\ [[J][\sigma]] + \{F\} = \{0\} \tag{2.4.14}$$

The above represents three scalar equations one for each direction {i}. The first equation is given by

$$\frac{\partial}{\partial x}(\Sigma J_{1j}\sigma_{j1}) + \frac{\partial}{\partial y}(\Sigma J_{1j}\sigma_{j2}) +$$

$$\frac{\partial}{\partial z}(\Sigma J_{1j}\sigma_{j3}) + F_x = 0 \qquad j = 1,2,3 \qquad (2.4.15)$$

2.5 Principle of Virtual Work

The principle of virtual work, or more precisely the principle of virtual displacements, provides an alternative but equivalent statement of the equilibrium conditions for a body. In the finite element method this principle is used to derive a finite number of nonlinear algebraic equations that approximate the nonlinear partial differential equations of equilibrium in Eqs. 2.4.5 and 2.4.14.

Virtual work principles are customarily derived first for a particle, a collection of particles, and finally for a continuum. Highlights of the derivation for the Lagrangian system are presented here since it is more complex than the Eulerian derivation. The algebraic operations are quite lengthy but relatively straightforward. Equilibrium equations of 2.45 are multiplied by arbitrary virtual displacements,$\{\delta U\}$, and integrated over the deformed volume. To this is added the integral over the deformed surface the product of virtual displacements times $\{\bar{T}'\}-\{T'\}$, where $\{\bar{T}'\}$ are components of specified surface traction and $\{T'\} = [\sigma']\{n'\}$. The virtual displacements are considered functions of the initial coordinates $\{X\}$ and satisfy all geometric boundary conditions imposed upon the body. The integration limits are transformed to the undeformed configuration by introducing the Lagrange stress components as in Eq. 2.47. Green's theorem is then applied to the surface integral transforming it into an equivalent integral over

the undeformed volume. All Lagrange stress components multiplied by variations of displacement gradients are grouped together, then the Lagrange stress components are expressed in terms of 2nd Piola-Kirchoff stresses defined by Eq. 2.4.11. The coefficients multiplying each component of 2nd Piola-Kirchoff stress are seen to be precisely the variations of Green strain components defined in Eq. 2.3.22. The remaining terms containing derivatives of Lagrange stresses multiplied by virtual displacements are re-expressed as a surface integral giving the virtual work of all surface tractions applied over the body. The resulting integrals provide the equilibrium equations

$$\int_V \{\delta U\}^T \{F\}\, dV = \int_V \{\delta \varepsilon_g\}^T \{\sigma\}\, dV - \int_S \{\delta U\}^T \{T\}\, dS \qquad (2.5.1)$$

The left most integral is the virtual work of all body forces acting through the virtual displacements. The middle integral represents the internal virtual work and the rightmost integral is the work of applied surface tractions, $\{T\}$, specified in directions of unit vectors $\{i\}$ with intensities always based on the initial area. Defining the work of body forces as δW_{bf}, the internal work δW_i as the negative of the second integral, and the work of surface tractions as δW_s, the equilibrium equations are

$$-\delta W_i = \delta W_{ext} = \delta W_{bf} + \delta W_s \qquad (2.5.2)$$

As will be shown in Chapter 3, Eq. 2.5.1 is directly applicable in the nonlinear finite element method.

The virtual work equations for the Eulerian formulation are identical to those above with the appropriate changes in components

$$\int_{V'} \{\delta U\}^T \{F'\}\, dV' = \int_{V'} \{\delta \varepsilon_a\}^T \{\sigma'\}\, dV' - \int_{S'} \{\delta U\}^T \{\bar{T}'\}\, dS' \qquad (2.5.3)$$

Variations of Almansi strain replace Green strain, Cauchy stresses replace 2nd Piola-Kirchoff stresses, surface tractions are for deformed areas and body forces are per unit deformed volume.

2.6 Stress-Strain Relations

The finite element method has provided new impetus for research efforts into the understanding of nonlinear material behavior. Before finite element technology was available, nonlinear analysis was complicated by material behavior as well as large deformations and irregular boundary conditions. The finite element method can approximate the latter two factors to within any accuracy desired but must still rely upon theoretical models of material behavior for stress-strain relations.

The purpose of a material model is to predict the stress at a point in the body given the strain. This may be accomplished in either total or incremental form as

$$\{\sigma\} = [D_S]\, \{\varepsilon\} \qquad (2.6.1)$$

$$\{\sigma\} = \Sigma\{\Delta\sigma\} = \Sigma[D_T]\{\Delta\varepsilon\} \qquad (2.6.2)$$

where $[D_S]$ is termed the secant modulus matrix and $[D_T]$ the tangent modulus.

Strains are either Green or Almansi with corresponding 2nd Piola-Kirchoff stresses or Cauchy stresses. For linear elastic, small displacement analysis $[D_S] = [D_T]$ and each matrix contains the elastic constants referred to the initial configuration.

When stresses exceed the proportional limit or strains become finite in magnitude, $[D_S]$ and $[D_T]$ are no longer given by the elastic constants and a nonlinear [D] matrix dependent upon current strain, previous loading history, etc. is required. Three basic situations arise:

1) Infinitesimal strain magnitudes with stresses that exceed the proportional limit of the material. Behavior is often adequately predicted by classical plasticity and fracture (cracking) theories.

2) Strains have finite magnitude but the material is highly elastic, for example rubber. Mooney-Rivilin models have been proposed with terms of $[D_S]$ given by differentiation of the strain energy density function (determined experimentally) with respect to Green strain components.

3) Finite strains in materials that exhibit yielding behavior thus requiring the application of large strain plasticity theory.

Fortunately, most nonlinear problems of interest to structural engineers involve the first type of behavior listed above with materials such as concrete and steel. Structures having this characteristic include frames, plates, shells, and most solid bodies. Frames, plates and shells may have large rotations under loading but small strains.

A general approach to nonlinear finite element analysis necessitates an incremental technique for solving the equilibrium equations; thus, only incremental forms of stress-strain relations of Eq. 2.6.2 are required. In the Lagrangian formulation, increments of 2nd Piola-Kirchoff stress can be related directly to increments of Green strain through the tangent modulus matrix for the material. No coordinate transformations are necessary as stresses are always based on undeformed areas and the direction of strain and stress components coincide with unit vectors {i}. However, Cauchy or Eulerian stresses based on projections of deformed areas and Almansi strain based on deformed coordinates cannot be directly related as the deformed area and coordinates change during the increment. For an increment consisting of pure rigid body rotation, projections of the deformed area onto coordinate planes change thus changing the Cauchy stresses. A technique to account for rotations during an increment was introduced by Jaumann who defined a stress increment $\{\Delta\sigma_J\}=[D_T]\{\Delta\varepsilon_a\}$ with the increment of Eulerian stress then given by

$$\{\Delta\sigma\} = \{\Delta\sigma_J\} + [\Delta T_\omega]\{\sigma'_{OLD}\} \qquad (2.6.3)$$

where $[\Delta T_\omega]$ is the matrix of incremental rigid body rotations due to the displacement increment. Thus, the change of stress is determined by a combination of rigid body rotation and additional straining. The $[D_T]$ matrix contains the usual elastic constants or terms resulting from classical plasticity. The terms of $[\Delta T_\omega]$ are given in Ref. [49]. Eq. 2.6.3 also shows that before Eulerian stress increments can be combined they must first be rotated to a common coordinate system then summed and transformed back to final deformed coordinates. This is usually accomplished by transforming to 2nd Piola-Kirchoff stress increments via Eq. 2.4.11.

An additional complication with the Eulerian formulation involves materials that are initially anisotropic. As the body deforms, principal material property directions constantly change; thus, the [D] matrix must be transformed at each increment in the analysis. This problem does not occur in the Lagrangian formulation.

2.7 Summary

In this chapter the basic theory of nonlinear continuum mechanics has been presented in a form suitable for finite element application. As shown, two uniquely different approaches, Lagrangian and Eulerian, are available to formulate nonlinear structural analyses.

Early attempts at nonlinear finite element analysis employed the Eulerian formulation as a natural extension of existing linear analysis systems. Coordinates of node points were simply updated by the incremental displacements and another linear analysis performed. However, as seen in this chapter, such an approach requires continual transformations of coordinates, strains, stresses, and material properties. This factor severely limits the efficiency of the Eulerian approach for large scale structural analyses. It has traditionally been most successful for applications in which bodies suffer extreme deformations under loading, such as a viscous fluid.

The Lagrangian formulation has none of these disadvantages and considerably simplifies the overall analysis. It is applicable for all common structures that do not exhibit severe distortions under loading. Therefore, the Lagrangian formulation was adopted in this study for implementation of a general nonlinear finite element system for structural analysis.

CHAPTER 3

NONLINEAR ASPECTS OF THE FINITE ELEMENT METHOD

3.1 General

The finite element method (FEM) has gained widespread acceptance among engineers for the analysis of linear structures. The development of elements capable of modeling most structural components combined with the intuitive and mathematical bases of the method has contributed to its popularity. Finite element techniques have also proven invaluable in nonlinear applications. Nonlinear analysis of practical structures is achieved by approximating the complex nonlinear partial differential equations of continuum mechanics with a set of nonlinear algebraic equations.

The following three classes of nonlinear structural response are considered in this work:

1) Small displacements, small strains with linear and nonlinear material properties;
2) Large displacements resulting in significant rotations but small strains with linear and nonlinear material properties.
3) Large displacements with associated finite strain magnitudes also with linear and nonlinear material properties.

The first category has received the most attention of finite element researchers and includes ordinary linear structures and solid mechanics problems with material nonlinearities. Elastic and inelastic stability analyses of frames, plates and shells fall into the second category. Problems in which finite strains occur are generally the concern of

engineers working in non-structural applications and involve materials such as rubber and plastic. Emphasis in this study was placed on the first and second classes of structures for which some constitutive relations are available for common construction materials. However, the computer system discussed in Chapter 5 based on the formulation presented here, is capable of solving finite strain problems if appropriate constitutive relations are provided.

This chapter briefly examines the finite element method for nonlinear analysis within the Lagrangian formulation. Full details of the method for the first class of problems listed above are available in most standard texts [13,15,73]. The discussion here concentrates on extensions of the basic formulation necessary to incorporate large displacement effects. Details of these techniques are not generally available in the literature. The response of isolated individual elements is discussed prior to procedures for generating and solving the nonlinear equilibrium equations. Special consideration is given to the residual load concept and the application of substructuring with static condensation.

3.2 Individual Elements

Once a structure is partitioned into a mesh of finite elements, individual elements may be isolated and studied independently. Nonlinear equations describing the response of a single element are considered in this section. Discussion is limited to formulations based on assumed displacement fields as these elements are most commonly employed in practical analysis. Strain-displacement relations and element equilibrium equations are derived following a discussion of interpolating functions.

A general three dimensional element in rectangular coordinates, as shown in Fig. 3.2.1, is employed to illustrate specific terms of the matrices involved.

3.2.1 Interpolation Functions

The conversion of discrete displacement component values specified at node points into continuous analytic functions of the element local coordinates constitutes the basic step of the displacement formulation. Displacement components (degrees of freedom, dof) at each node are listed in an M x 1 vector, denoted $\{U_e\}$, where M is the total number of dof for the element nodes. Displacements at points P(x,y,z) within the element as functions of local element coordinates are given by

$$\{U(x,y,z)\} = \sum_{1}^{nn} [N_i(x,y,z)] \{U_e\}_i \qquad (3.2.1)$$

where nn is the number of element nodes, $[N_i]$ is a matrix of interpolating or "shape" functions for the i^{th} node, and $\{U_e\}_i$ are displacement components for the i^{th} node. For a three dimensional element with dof U, V, and W at each node, the above equation is

$$\begin{Bmatrix} U(x,y,z) \\ V(x,y,z) \\ W(x,y,z) \end{Bmatrix} = \sum_{1}^{nn} \begin{bmatrix} N_i & 0 & 0 \\ 0 & N_i & 0 \\ 0 & 0 & N_i \end{bmatrix} \begin{Bmatrix} U \\ V \\ W \end{Bmatrix}_i \qquad (3.2.2)$$

Each component of displacement above is interpolated over the element with identical shape functions. In general, different dof can be defined at each node thereby forming a variable dof element. Such an element proves useful to transition between two different classes of elements, for example, shells (6 dof/node) and three dimensional solids (3 dof/node).

Other quantities for the element may also be interpolated with shape functions. For example, the thickness of planar elements, varying material properties, temperature, etc. are easily converted to functions of element coordinates by interpolation from specified nodal values.

3.2.2 Strain-Displacement Relations

Before the element equilibrium equations can be derived, it is first necessary to determine the variation of Green strain corresponding to a variation of nodal displacements. From Eq. 2.3.21 the required relation can be written

$$\{\delta\varepsilon_g\} = [B]\ \{\delta U\} \tag{3.2.3}$$

where $\{\delta\varepsilon_g\}$ is a 6 x 1 vector containing the variation of Green strain components as functions of the undeformed element coordinates. [B] is the 6 x 3 differential operator matrix defined in Eq. 2.3.22. $\{\delta U\}$ is the 3 x 1 vector of local element displacement variations defined in terms of nodal displacement variations by

$$\{\delta U\} = \sum_{1}^{nn} [N_i]\ \{\delta U_e\}_i \tag{3.2.4}$$

Derivatives of the deformed coordinate system $\{X'\}$ appearing in terms of [B] are obtained by differentiating

$$\{X'\} = \{X\} + \sum_{1}^{nn} [N_i]\ \{U_e\}_i \tag{3.2.5}$$

Similarly, derivatives of displacements are given by differentiating Eq. 3.2.1 Substituting derivatives in terms of shape functions into Eq. 2.3.22 yields the variation of Green strain in terms of finite element quantities for the three dimensional case as

$$\{\delta\varepsilon_g\} = \sum_{1}^{nn} [B_i] \{\delta U_e\}_i \qquad (3.2.6)$$

where the terms of $[B_i]$ are

$$[B_i] = \begin{bmatrix} X'_x N_x & Y'_x N_x & Z'_x N_x \\ X'_y N_y & Y'_y N_y & Z'_y N_y \\ X'_z N_z & Y'_z N_z & Z'_z N_z \\ X'_y N_x + X'_x N_y & Y'_y N_x + Y'_x N_y & Z'_y N_x + Z'_x N_y \\ X'_x N_z + X'_z N_x & Y'_x N_z + Y'_z N_x & Z'_x N_z + Z'_z N_x \\ X'_z N_y + X'_y N_z & Y'_z N_y + Y'_y N_z & Z'_z N_y + Z'_y N_z \end{bmatrix} \qquad (3.2.7)$$

Derivatives of shape functions in the equation refer to the function for the i^{th} node, N_i. For small displacement, linear analysis, $X'_x = Y'_y = Z'_z = 1$; other derivatives of $\{X'\}$ are zero. $[B_i]$ then simplifies to the well known matrix employed in linear analysis. However, in large displacement analysis, the [B] matrix is a function of the nodal displacements because derivatives of deformed coordinates do not vanish. By imposing the Kirchoff plane sections hypothesis, specific forms of the [B] matrix are obtained for frames, plates, and shells.

An expression for the total Green strain follows readily from terms of the [B] matrix. First, separate $[B_i]$ into the sum of two matrices, $[B_i^L]$ and $[B_i^{NL}]$, such that $[B_i^L]$ represents the usual linear matrix with $[B_i^{NL}]$ containing the nonlinear terms. A simple re-examination of the expanded total Green strain components, Eq. 2.3.10, shows that the total strain is given by

$$\{\varepsilon_g\} = \sum_{1}^{nn} [[B_i^L] + \frac{1}{2} [B_i^{NL}]] \{U_e\}_i \tag{3.2.8}$$

3.2.3 Element Loads

Loads applied over an element are expressed as functions of local element coordinates by interpolation of nodal values. Intensities of body force components, {F}, are given by

$$\{F\} = \sum_{1}^{nn} [N_i] \{F_e\}_i \tag{3.2.9}$$

where nodal values are force per unit undeformed volume.

Components of a traction applied over the element boundaries are interpolated from traction intensities at nodes such that

$$\{T\} = \sum_{1}^{nn} [N_i] \{T_e\}_i \tag{3.2.10}$$

where $\{T_e\}_i$ is the 3 x 1 vector of traction components at the i^{th} node referred to initially undeformed areas and element local coordinate directions.

3.2.4 Element Virtual Work Equations

All components required to write the virtual work equation in the Lagrangian formulation, Eq. 2.5.1, are now available in terms of finite

element quantities. By substituting Eqs. 3.2.4 and 3.2.9 into Eq. 2.5.1, the virtual work of body forces is seen to be

$$\delta W_{bf} = \sum_{1}^{nn} \int_V \{\delta U_e\}_i^T [N_i]^T [N_i] \{F_e\}_i \, dV \tag{3.2.11}$$

A simpler form of this equation is obtained by using an implied summation over the element nodes and by removing nodal quantities independent of the coordinates under the integral. Thus, Eq. 3.2.11 simplifies to

$$\delta W_{bf} = \{\delta U\}^T \left[\int_V [N]^T [N] \, dV \right] \{F\} \tag{3.2.12}$$

where integration is over the initial volume of the element.

Similarly, the virtual work of applied element surface tractions is given by

$$\delta W_{st} = \{\delta U\}^T \left[\int_S [N]^T [N] \, dS \right] \{T\} \tag{3.2.13}$$

with integration performed over the undeformed element surface.

The total virtual work of applied element loads is then

$$\delta W_{ext} = \{\delta U\}^T \{P\} \tag{3.2.14}$$

where,

$$\{P\} = \left[\int_V [N]^T [N] \, dV \right] \{F\} + \left[\int_S [N]^T [N] \, dS \right] \{T\} \tag{3.2.15}$$

$\{P\}$ contains the equivalent (consistent) nodal loads for the element.

Nonlinearities enter Eq. 3.2.15 in large displacement analysis when significant differences exist between the initial and deformed element geometry. In such cases, body force and surface traction components

at nodes specified in the deformed geometry must be transformed to the initial geometry before evaluation of Eq. 3.2.15.

The internal virtual work is simply

$$\delta W_{int} = -\{\delta U\}^T \{\int_V [B]^T \{\sigma\}\, dV\} \tag{3.2.16}$$

in finite element terms. This can also be written as

$$\delta W_{int} = -\{\delta U\}^T \{IF\} \tag{3.2.17}$$

where {IF} is often called the element internal nodal force vector. Terms of {IF} are components of force, in the generalized sense, exerted on the element by the nodes due to the element's state of deformation. Nonlinearities enter through the [B] matrix for large displacements and through $\{\sigma\}$ for nonlinear material behavior.

3.3 Structure Equilibrium Equations

The structural equilibrium equations in terms of nodal forces result from a straightforward application of statics at each node. In summing forces at each structural node, element internal forces and equivalent nodal loads require rotation from element local to structural global coordinate frames and proper placement in the structure equilibrium equations. This is accomplished in the usual way with an element rotation matrix, $[\lambda_i]$, and the Boolean connectivity matrix, $[L_i]$, such that the structural equilibrium equations are generated symbolically as

$$\{R\} = \sum_1^N [L_i]^T [\lambda_i]^T \{P\}_i - \sum_1^N [L_i]^T [\lambda_i]^T \{IF\}_i \tag{3.3.1}$$

or more simply,

$$\{R\} = \{P_S\} - \{IF_S\} \tag{3.3.2}$$

where $\{P_S\}$ and $\{IF_S\}$ are the total structure applied load vector and total internal force vector respectively. The summation over all elements also includes substructures in addition to simple finite elements without any changes in notation. The difference between applied load and internal force vectors, termed the residual nodal loads {R}, must vanish for equilibrium.

Eq. 3.3.1 shows a major computational advantage of the Lagrangian formulation. The rotation matrix, $[\lambda_i]$, for each element is independent of the nodal displacements. In the Eulerian approach, every modification of nodal displacements necessitates recomputation of the rotation matrices.

3.4 Solution of Equilibrium Equations

Each term of the residual nodal load vector {R} is indirectly a nonlinear algebraic function of the nodal displacement components {U}. A set of nodal displacements corresponding to an equilibrium configuration satisfies the relation

$$\{R(\{U\})\} = \{0\} \tag{3.4.1}$$

Solution of the nonlinear finite element problem requires solution of 'n' simultaneous nonlinear algebraic equations where 'n' is the total number of unknown nodal displacement components.

Of the many techniques developed for solving nonlinear equations, those based on the Newton-Raphson method are the most popular and widely known. The Newton-Raphson method was chosen for use in this work because of its applicability to problems involving load path dependent material behavior, finite elastic deformations, and to problems with both geometric

and material nonlinearities. The method is by no means perfect; cases exist where it diverges. Two examples are "locking" structures and structures that exhibit "snap-through" buckling, i.e., the solution process cannot trace the unstable portion of the load-deflection curve. However, it readily solves equations arising in the majority of practical nonlinear analyses. A brief description is included herein; additional details are given in Ref. [15].

3.4.1 The Newton-Raphson Method

The Newton-Raphson process forms the basis of several proposed solution strategies. The technique adopted in this work, termed incremental-iterative, is the most general and contains other forms as special cases. The total load, $\{P_S\}$, applied to the structure is divided into a number of increments or load steps $\{\Delta P_S\}$. The number and size of load steps chosen for analysis is usually based on the severity of the nonlinear response and the load levels at which results (displacements, stresses, and strains) are desired. The solution proceeds in a stepwise fashion for successive load steps beginning with step one. Changes of nodal displacements within a load step are computed by a series of linear approximations (iterations). The correct displacements are assumed to have been reached when, for example, terms of the residual load vector, Eq. 3.3.2, are sufficiently small. Solution for the next step is then begun.

A major component of the Newton-Raphson solution process involves correcting the nodal displacements to eliminate the residual loads. For a given set of nodal displacements, $\{U\}$, that do not satisfy the requirement of $\{R\} = \{0\}$, a correction $\{dR\}$ is sought such that

$$\{R\} + \{dR\} = \{0\} \tag{3.4.1}$$

A differential change in the residual nodal loads is given by

$$\{dR\} = \{dP_S\} - \{dIF_S\} = [J_S]\ \{dU\} \tag{3.4.2}$$

where the Jacobian matrix, $[J_S]$, contains partial derivatives of the function $\{R\}$ with respect to nodal displacements evaluated at the current displacements. The Jacobian is conveniently split into the sum of two matrices such that

$$\{dR\} = [K]\{dU\} = [\ [K_L] - [K_T]\ ]\ \{dU\} \tag{3.4.3}$$

$[K_L]$ is the "initial load stiffness" and accounts for changes in displacement dependent loadings (nonconservative loads). The matrix $[K_T]$, termed the "tangent stiffness matrix", determines changes in internal nodal forces corresponding to changes in nodal displacements. Eq. 3.4.3 represents a set of linear simultaneous equations linking differential changes in the residual load vector and nodal displacements. Substitution of Eq. 3.4.3 into 3.4.1 and changing from differential to finite increments yields an expression for the corrective nodal displacements that eliminates the residual load

$$\{\Delta U\} = -\ [K]^{-1}\{R\} \tag{3.4.4}$$

Solution of the linear equations is performed by Gaussian elimination or some other triangulation scheme rather than formal inversion as indicated above.

A graphical summary of the various algorithms based on the Newton-Raphson procedure for a single nonlinear equation is given in Fig. 3.4.1. In the classical formulation, illustrated in Fig. 3.4.1b, [K] is regenerated and triangulated before each iteration to assure the best possible convergence rate. However, this procedure is prohibitively expensive for structures with many nodes. In a modified version of Newton-Raphson, [K] is reformed and triangulated before each step but held constant for all iterations within the step as shown in Fig. 3.4.1c. More iterations are required but the cost of each iteration is drastically reduced. In the limiting case, the conventional linear stiffness matrix is used for all load steps (Fig. 3.4.1d). This modification, termed the "constant stiffness method", converges only for problems with slightly nonlinear behavior, and even these may require an excessive number of iterations. Still other forms of Newton-Raphson update the stiffness matrix before specific iterations of a load step in an attempt to optimize the convergence rate. But as is evident, all the various forms are special cases of the original Newton-Raphson method and are easily incorporated into a general solution algorithm.

3.4.2 Outline of Solution Procedure

The following is a summary of the major computational steps required to analyze the structure for each load step:

1) Compute the equivalent nodal loads, $\{\Delta P_S\}$, corresponding to the increment of applied loads defined for the step. Set $\{R\} = \{\Delta P_S\}$.
2) If required, update the [K] matrix and triangulate.

3) Update the total nodal loads currently applied to the structure.

4) Solve for an increment of the nodal displacements using the current triangulated stiffness $\{\Delta U\} = [K]^{-1} \{R\}$.

5) Compute increments of Green's strain and the new total 2nd Piola-Kirchoff stresses for each nonlinear element. For large displacement analysis, $[B_i]$ is evaluated at $\{U\} + \frac{1}{2} \{\Delta U\}$ to compute the correct increment of strain.

6) Update the total nodal displacements, element strains, and stresses to reflect the previously computed displacement increment.

7) Evaluate the internal nodal forces for all elements using current total displacements and stresses. For linear elements and substructures $\{IF\} = [K]\{U\}$ where $[K]$ is the conventional linear stiffness matrix. For nonlinear elements, internal forces are computed by Eq. 3.2.16.

8) Compute the structure residual nodal load vector as $\{R\} = \{P_S\} - \{IF_S\}$.

9) Apply convergence tests on $\{R\}$, $\{U\}$, or $\{\Delta U\}$. If the convergence test(s) are satisfied, go to step (1) and begin processing the next step; otherwise go to step (10).

10) Update and triangulate the stiffness matrix if required. Then return to step (4) to begin the next iteration.

3.4.3 Convergence Tests

An important aspect of the nonlinear solution process involves selection of suitable criterion to indicate the iteration process has

converged to a sufficiently accurate solution. The Euclidean norm (square root of the summed squares) of the nodal displacement and residual load vectors comprise the most popular convergence test quantities. Four common tests are listed below:

1) $\| \{R\} \| < \| \{\Delta P\} \| * \text{tolerance}$

2) $|\max \{R\}_i| < \| \{\Delta P\} \| * \text{tolerance}$

3) $\| \{\Delta U\}_i \| < \| \{\Delta U\}_1 \| * \text{tolerance}$

4) $|\max \{\Delta U\}_i| < \| \{\Delta U\}_1 \| * \text{tolerance}$

where the subscript refers to the iteration number.

The first test simply compares lengths of the 'n' dimensional applied load and residual load vectors. The second test compliments the first by detecting any highly localized residual loads that would be smoothed over by the vector norm computation. These two tests are most applicable to solid mechanics problems in which all generalized forces are of the same class. In frame, plate, and shell structures, generalized forces consist of moments, shears, and membrane forces which often vary by several orders of magnitude in size. The first two convergence tests are less appropriate for these types of structures.

The third and fourth tests are based on the reduction of corrective displacement magnitudes as iterations proceed for a load step. Both assume that large changes in displacement occur during the first few iterations and that changes become successively smaller in subsequent iterations.

The selection of tolerance values continues to be a matter of engineering judgment. The convergence test and tolerance value chosen tend to be highly problem dependent. Often, a nodal displacement and residual force criterion are specified, both of which must be satisfied

for convergence. Hand [28] used a test similar to the fourth one above with a tolerance of 5%. Nayak and Zienkiewicz [50] relied on tolerances ranging from 1% to 0.01% for the first test with the total load norm substituted for the incremental load norm. This latter test based on the total load vector norm can be misleading. As the applied load on the structure increases, the total load norm increases quadratically and for a constant tolerance the overall convergence criterion becomes less severe especially during the final stages of analysis when nonlinear behavior is most pronounced.

During this study numerous structures were analyzed using the convergence criterion outlined above. The residual load criterion proved useful as a guide for selecting subsequent load step sizes. It was also found that unduly small tolerances produced only minor improvements in the solution. For example, differences in solutions were insignificant for 1% and 0.1% tolerances with the first convergence test.

3.5 Element and Structure Stiffnesses

Expressions for the element tangent stiffness matrix $[K_T]$ are derived from the internal forces

$$\{IF\} = \int_V [B]^T \{\sigma\}\, dV \tag{3.5.1}$$

where both $[B]$ and $\{\sigma\}$ are indirectly functions of the element nodal displacements. Differentials of internal forces are then

$$\{dIF\} = \int_V d[B]^T \{\sigma\}\, dV + [B]^T \{d\sigma\}\, dV \tag{3.5.2}$$

Expansion of the second integral using the material tangent modulus matrix, Eq. 2.6.2 yields

$$\int_V [B]^T \{d\sigma\}\, dV = \left[\int_V [B]^T [D_T][B]\, dV \right] \{dU\} \qquad (3.5.3)$$

For small displacement, linear and material nonlinear analysis, the [B] matrix is independent of nodal displacements (contains only element nodal coordinates); thus, the first integral of Eq. 3.5.2 vanishes leaving the familiar second integral as the element tangent stiffness matrix. For geometric nonlinear analysis the first integral must be included. Unfortunately, the expansion of $d[B]^T\{\sigma\}$ requires considerable manipulation to obtain a useful form. Nayak [49] was the first to present a matrix formulation for a general three dimensional element and his notation is followed here. A different and more detailed derivation of the matrix formulation is given in Appendix A. As shown in the Appendix, the first integral of Eq. 3.5.2 can always be written in the form

$$\int_V d[B]^T\{\sigma\}\, dV = \left[\int_V [G]^T [M][G]\, dV \right] \{dU\} \qquad (3.5.4)$$

where [G] contains only element shape function derivatives and [M] is a symmetric matrix of stresses within the element. The above integral generates what is commonly termed the "initial stress" stiffness matrix. The element tangent stiffness matrix containing all geometric and material nonlinear terms can be written as

$$[K_T] = \left[\int_V [G]^T [M][G] + [B]^T [D_T][B]\, dV \right] \qquad (3.5.5)$$

The tangent stiffness matrix above is symmetric for a symmetric $[D_T]$ matrix.

The initial load stiffness matrix, $[K_L]$, for an element reflects changes in the equivalent nodal loads for displacement dependent applied

loadings. Such loads include pressures on highly deformable membranes in which significant changes of area magnitude and direction occur. This relationship is expressed by

$$\{dP\} = [K_L]\,\{dU\} \tag{3.5.6}$$

Nayak has shown that $[K_L]$ is generally a nonsymmetric matrix with terms quite small compared to $[K_T]$. Details of the derivation follow the same procedure as used to derive $[K_T]$. The nonsymmetry is anticipated by noting that area transformations between initial and deformed configurations derived in Chapter 2 are nonsymmetric. The nonsymmetry of $[K_L]$ considerably complicates determination of corrective displacements in the solution process by making the structure stiffness nonsymmetric. Nonsymmetry of the equations essentially doubles the solution effort required. The smallness of terms in $[K_L]$ combined with its nonsymmetry and the fact that in most structures loads are independent of displacement suggest neglecting $[K_L]$ in computing the element stiffness. The effect of any nonconservative loads is then accounted for during computation of equivalent nodal loads at the beginning of each load step using the deformed geometry.

In the Lagrangian formulation, once element stiffness matrices are computed, assembly of the structure stiffness matrix follows the direct stiffness method as for linear analysis. This process is described in all standard texts.

3.6 Substructuring and Static Condensation

Substructuring and static condensation are natural extensions of the basic finite element analysis method. The analyst divides the structure

into substructures along convenient internal boundaries separating logical components of the structural model. Substructures may be further subdivided into simple finite elements or into other substructures continuing for as many levels as practical. Typically, each substructure models some portion of the actual structure that appears repeatedly in the next higher level substructure.

A hierarchial tree as shown in Fig. 3.6.1 provides a graphical representation for a system of substructures. The final structure is the root "node" of the tree; substructures appear as intermediate nodes. Simple finite elements must compose all terminating nodes of the tree, i.e., nodes with no lower "branches". Assembly of element and substructure stiffness matrices and load vectors begins with the terminating nodes and proceeds upward through the hierarchy. Solution for the nodal displacements is performed only for the highest level structure. Nodal displacements within substructures are determined by a mapping process beginning at the highest level structure and proceeding down the tree.

In linear analysis, substructuring eliminates the wasteful computation of stiffness matrices and equivalent nodal loads for identical elements and substructures (those appearing at more than one location in the tree). However, all nodes in lower level substructures also appear in higher level structures. The final structure has the same number of nodes as a model without substructuring.

The number of nodes in the highest level structure can often be drastically reduced through static condensation of lower level substructures. Consider, for example, the arbitrary structure shown in Fig. 3.6.2. Nodes of each substructure are classified as either interior or exterior.

Interior nodes are defined as nodes which do not directly connect to nodes of other substructures. Static condensation of a substructure produces a mathematically equivalent stiffness matrix and load vector for the exterior nodes that contain all effects of interior nodes. The actual substructure is thus replaced in the hierarchy by an equivalent substructure with only exterior nodes. Przemieniecki [57] describes the necessary computations in detail. The primary difference between the classical formulation given in the reference and the one used in practice is that no matrix inversions are performed. Instead special triangulation schemes based on one of the standard decomposition algorithms are employed.

Condensed substructures are introduced into the hierarchy between the substructure to be condensed and the substructure in which it appears as an element. Thus, a node in the tree representing a condensed substructure has only one lower level connecting branch as shown in Fig. 3.6.1. When a condensed substructure is encountered during stiffness or load vector assembly, the process is temporarily suspended while the condensation is performed. The normal assembly process then resumes. Viewed in this manner, condensation at any number of levels presents no difficulties in the formulation. Displacements inside condensed substructures are obtained by the process known as back-condensation during traversal down the tree after solution of the highest level structure.

A flexible substructuring and condensation capability enables the analyst to greatly reduce the computer cost incurred during a nonlinear analysis. Linear regions of the structure, often determined by inspection, are substructured and condensed leaving only nodes that interact with nonlinear substructures. Reductions in computer time result if a significant number of nodes can be eliminated from the incremental-iterative

nonlinear solution process. Savings of computer time occur primarily during the following solution phases:

1) Tangent stiffness update. Only stiffnesses of nonlinear elements and nonlinear substructures must be recomputed and assembled. Stiffness matrices of linear elements, substructures, and condensed linear substructures computed during the first load step, are retained for use throughout the analysis.

2) Triangulation of the tangent stiffness. Most nonlinear analyses require numerous updates and triangulations of the tangent stiffness of the highest level structure. The cost of analysis is generally dominated by the time required to triangulate the equations. Condensation of linear substructures eliminates many nodes from the highest level structure thereby reducing the effort required to repeatedly triangulate the equations. The time required to perform the initial condensation of linear substructures is generally trivial compared to the savings achieved in triangulations of the nonlinear structure stiffness. As the frequency of stiffness updates increases, for example, near the ultimate load, the efficiency of the solution with condensation greatly increases.

3) Strain and stress computations. Since the behavior of a linear element is not a function of the strains or stresses, it is not necessary to compute this data for linear elements, substructures, and condensed substructures during the nonlinear incremental solution. Displacements, strains, and stresses inside linear substructures are computed only at the user's request after the nonlinear solution has converged.

4) Residual loads calculation. The internal force vector for linear elements and substructures is computed from the product of the stiffness matrix and total displacements. This is particularly effective for condensed substructures as only the internal force vector for the boundary nodes is required. Residual loads are zero inside linear substructures by definition.

CHAPTER 4

REQUIREMENTS OF NONLINEAR FINITE ELEMENT SOFTWARE

4.1 General

The matrix formulation presented in the preceeding two chapters provides a description of the numerical procedures for solving nonlinear finite element problems. Implementation of this formulation into a software system for general purpose analysis is a formidable task. For example, man-machine communication, structural modeling, analysis restart, error recovery, computer resource utilization, flexibility, maintainability, and portability are a few of the difficulties faced by software engineers. Although these factors are not directly concerned with formulating or solving the governing equations, they generally determine the success or failure of the software system. These topics are a sharp contrast to the software problems of previous generations, when the primary concern was developing new algorithms suited for digital computers.

In the present chapter, only those aspects of nonlinear finite element software directly affecting the structural analyst are examined. Particular attention is given to the features and capabilities needed for successful large scale, general purpose nonlinear finite element software. Specific techniques employed to implement the type of software discussed here are presented in Chapter 5.

4.2 User-Program Interface

The most commonly overlooked aspect of engineering software involves communication between the user and the computer program. Input and output are usually viewed by program developers as nuisances to be hastily dealt with

They fail to recognize that a user experiences only three aspects of a computer system. These are a) data entry, b) output capability, and c) costs. Most users are willing to pay extra for a program that has flexible input and output features.

Traditionally, finite element programs have been executed in batch mode; data entry has generally been fixed format. Fixed format input is tedious and prone to errors. Checking is difficult because no descriptive labels appear within the data. Even experienced users must often refer to program manuals for data ordering and format fields.

Widespread use of timesharing computers for interactive processing prompted development of the so called question-answer data entry mode, in which the user is asked to respond to a series of questions via a keyboard. The primary objection to this type of input is that the user can only respond to the data requests presented by the program. Difficulties arise when mistakes are made. No convenient techniques exist with question-answer input to request that a particular question be re-asked. The only advantage of this mode of data entry is the ease with which it is programmed.

Interactive graphics is becoming a popular means of data entry for describing finite element models. Generally, a combination of two techniques is used. In the first, a menu is placed on the screen from which the analyst may select one or more items via a light pen or through cross hair alignment and function buttons. Based on the item selected, the user is shown another more detailed menu or he is asked to answer a series of questions through a standard keyboard. Although requiring considerably less effort on the analyst's part, these techniques are essentially a graphical form of the typical question-answer input mode.

The ideal mechanism for user-program interface in batch, interactive, and graphic environments would be voice communication via natural languages. Unfortunately, natural language translators are in the early research stage and are many years away from practical application. In the interim, artificial languages, known as Problem Oriented Languages (POLs), are being developed by software engineers. POLs are simple languages designed for specific engineering disciplines. They consist of easily remembered, English-like phrases. POLs permit users to "tell" the program what to do, placing the user on the offensive, as opposed to his defensive position in the question-answer mode of data entry.

POLs differ from fixed format and question-answer input in a number of desirable ways. First, commands are always free form; words and data items may be placed anywhere on an input line. Secondly, most well designed POL systems permit any logical ordering of commands and thoroughly check data for consistency during the input phase of solution. Most often, corrections can be made immediately in an interactive environment by simply re-entering the command.

Some engineers have expressed concern that translation of POLs is inefficient and increases computer costs. This is true only for analysis of very small linear structures. Experience has shown that for practical linear analyses and for all nonlinear analyses, numerical computations are the dominating factor in computer charges -- the cost incurred translating the data is trivial.

In large linear analysis systems, batch operation with fixed format data has always been standard. However, considerable interaction between the analyst and the software is mandatory for efficient nonlinear analysis.

Engineers must be more intimately aware of the structural response and reflect this knowledge in parameters controlling the solution process. Blind "number crunching" as often occurs in linear analysis is not appropriate. To support this type of interaction, the software must permit a variety of input modes -- batch, graphical, and interactive. Batch and graphical input through a POL is suitable for specifying the structural topology and loads. Multiple analysis restarts via interactive sessions supported by a POL are appropriate for requesting nonlinear analysis, output, and modifying solution parameters.

4.3 Structural Modeling

Developing an appropriate finite element model for a structure and specifying that model to the computer program is a time consuming task for structural analysts. Finite element software should provide features to permit adequate modeling of the nonlinear structure with a minimum of effort. Several aspects of this process are discussed in the following sections.

4.3.1 Substructuring and Static Condensation

Substructuring and static condensation have proven effective in reducing input data and computer costs associated with linear analysis [4,17]. Even greater savings are possible in nonlinear analysis when geometric and material nonlinearities can be isolated in specific regions of a structure. Linear regions, usually determined by inspection, are substructured and condensed leaving only boundary nodes that interact with nonlinear substructures. The result is a drastically reduced number of nodes present in the highest level structure on which the incremental-

iterative solution is performed. After the nonlinear solution converges for the highest level structure, displacements, strains, and stresses inside condensed lower level linear substructures need be computed only if requested by the analyst. Substructuring and condensation as just described are demonstrated in two example problems in Chapter 6.

The bulk of input data describing a linear or nonlinear structure consists of nodal coordinates and element incidences that define the size, orientation, and connectivity of elements. Substructuring, combined with sophisticated mesh generators, reduces the amount of input data for structures that contain repeated identical components. A substructure is defined to encompass all elements and nodes of a common structural component. Nodal coordinates in other occurrences of the same substructure are not required input. Only the incidences and orientation of the substructure as it appears in the higher level structure are necessary [34, 35]. Rotation angles are a convenient means of describing a substructure's orientation.

Similarly, various patterns of loads applied to nodes and elements of a substructure are defined only once. Sets of effective nodal loads may be selectively applied to each occurrence of the substructure in a higher level structure to model the real applied loads. The orientation of loads on each occurrence normally follows that of the substructure; however, an engineer can potentially specify load orientation independently of substructure orientation. A simple example involves gravity type loads in which the direction remains constant even though the substructure orientation is altered.

For maximum effectiveness, substructuring and condensation must be an integral feature of the structural definition process from the user's viewpoint. Once defined, a substructure used in another structure should appear as any other finite element to the analyst during input. Computational details, such as mapping nodal degrees of freedom between substructures, should be handled completely by the software. The analyst should not be required to specify the order and kind of operations necessary to effect multiple level condensations or to specify structural data that must be recomputed during a nonlinear analysis. No currently available nonlinear analysis system provides a completely automated substructuring and condensation capability. Only a few linear systems offer any type of substructuring. Implementation of these features greatly increases program complexity and development time if multi-level condensations with loads are permitted. In spite of the obvious complexities introduced into the software to support substructuring and condensation, these features appear essential for efficient solution of large nonlinear structures.

4.3.2 Loads

Loading conditions defined for linear analysis generally consist of nodal forces, element pressures, temperature gradients, etc. that correspond to some real loading situation. In nonlinear analysis, the concept of a loading condition is extended to imply a loading "pattern" which defines a spatial distribution of loads on a structure. A single nonlinear load step may consist of any number of loading patterns combined with scalar multipliers to form the real load increment. To incorporate time dependent effects, such as creep and shrinkage, the analyst must be able to associate a time parameter with the definition of a nonlinear load step.

Multiple load patterns in a load step are essential. For instance, analysts may want to simulate a construction sequence by applying dead loads followed by live loads in different steps. Similarly, certain loadings may be simpler to describe when broken into components and applied as different loading patterns. Systems that permit only one pattern in a load step limit the analysis to proportional loading.

Loading definition becomes more complex with the introduction of multi-level substructuring and condensation. The following technique developed during this study appears to be quite flexible.

As lower level substructures in the hierarchy are defined, the declaration of loading patterns is performed as for linear loading conditions. In higher level substructures, loading patterns are defined in terms of patterns on lower level substructures. (Lower level substructures are treated as elements in higher level structures during structural definition). Finally, any number of loading patterns may exist on the highest level structure. A special type of loading condition, designated as nonlinear, is declared only for the highest level structure. A nonlinear loading condition specifies individual step load increments consisting of multiples of pattern loads on the highest level structure with any additional time parameters.

During solution for a load step, the equivalent nodal loads for each pattern specified in the step are computed and combined with the specified multipliers. For loading patterns that appear in more than one step, the equivalent nodal loads need to be computed only once and saved between load steps. However, equivalent nodal forces for loading patterns on nonlinear substructures must be recomputed before each step to account for

nonconservative effects. The software should automatically determine the appropriate sequence of operations (including condensation of loads) to effect the proper solution without intervention by the analyst.

4.3.3 Constraints

The tangent stiffness matrix of a structure contains rigid body motions that prevent a unique solution for displacements. A sufficient number of constraint equations to remove all possible rigid body motions are necessary before the equilibrium equations are rendered non-singular. Any number of additional consistent and nonredundant constraints are permitted to model the physical boundary conditions of the structure.

Two classes of constraints are necessary for general analysis. The first class, termed absolute constraints, are of the form

$$U_i = \text{constant} \tag{4.3.1}$$

and force the i^{th} degree of freedom displacement to equal the constant. The second class, termed relative constraints, force a linear relationship between displacements of two or more degrees of freedom. Such constraints are expressed by

$$\Sigma \alpha_i U_i = \text{constant} \tag{4.3.2}$$

Relative constraints are useful for connecting various types of finite elements, for instance, beam and shell elements.

During a nonlinear incremental solution, constraints also are incremental. Thus, absolute and relative constraints for nonlinear analysis are expressed by

$$\Delta U_i = \text{constant} \tag{4.3.3}$$

$$\Sigma \alpha_i \Delta U_i = \text{constant} \tag{4.3.4}$$

A special case occurs when the constant is always zero. Degrees of freedom with absolute constraints then always have zero displacements. With relative constraints, the relationship between displacements remains unaltered throughout the analysis.

The interpretation of constraints with non-zero constants must be incremental. During the first iteration of a load step, corresponding to an increment of applied loads, incremental displacements are forced to satisfy the constraint equations. If further iterations are necessary for a step to correct the linearized displacements, the constraint equation constants should be made temporarily zero. Otherwise, at completion of the iterations the sum of displacement changes occurring over the step would not satisfy the specified constraint equations for the step.

4.4. Solution Algorithms and Convergence Criterion

The Newton-Raphson procedure is at present the most widely applicable method for solving the nonlinear finite element equilibrium equations. The procedure has a very simple physical interpretation and converges relatively fast for most problems. Numerical analysts have not found the method particularly appealing because a good initial estimate of the solution is required. However, this is seldom a problem in structural applications as the previously converged nonlinear solutions provide a good approximation for subsequent load steps.

Several forms of the Newton-Raphson method have been developed; all are essentially the same algorithm as discussed in Chapter 3. A general system must provide the analyst with at least the options shown in Chapter 3. The algorithm is controlled by essentially two parameters:

a) The frequency of tangent stiffness updates;

b) The number of iterations per step to improve the solution and the criterion for terminating the iterative process.

The analysis of some large structures may necessitate the use of condensed nonlinear substructures to overcome software size limitations. Extension of the iterative technique and convergence criterion requires additional consideration because both the nonlinear substructures and the highest level structure have distinct residual loads and corrective displacements (no corrections are required in linear substructures). Residual loads for individual nonlinear substructures are also reflected in residual loads of higher level substructures as a result of the condensation process. To assure convergence of the nonlinear solution in all parts of the structure, the Newton-Raphson convergence tests should be performed on all condensed nonlinear substructures as well as the highest level structure. The solution is said to converge when the tests are satisfied in all substructures.

4.5 Analysis Restart and Error Recovery

A general restart capability is mandatory for all forms of nonlinear analysis. The analyst is seldom able to specify, a priori, all loading increments, convergence parameters, output requests, etc. Users typically will discontinue the analysis after several load steps, examine structural behavior for the current load level, modify data for subsequent steps, and then resume analysis in another computer run. This process may be repeated numerous times for complicated structures.

Associated with restart capability is the ability to recover from various errors during solution. Suppose, for example, that the structure

suddenly becomes unstable and the solution diverges after several load increments. If large load increments are used, the critical load of the structure is not well defined. In this situation, the analyst redefines the last load step, declares additional load steps, and requests a resumption of the analysis from the last converged step. To accomplish this, a system need only retain the appropriate data to resume at any step based on the type of nonlinearities present in the structure. Retaining all computed data for each step wastes space on secondary storage devices. Analysts should be permitted to request destruction of data for steps no longer needed, thereby releasing the space for reuse.

Certain parts of a structural description, for example, incidences and coordinates, cannot be changed during a nonlinear solution. However, users should be able to easily modify those aspects of the structural description and solution parameters listed below:

a) Loading definitions -- new loading patterns and load step increments;

b) Convergence parameters -- this includes both the types of convergence tests and the tolerances;

c) Solution algorithm -- the solution algorithm is controlled by the frequency of stiffness updating and the number of equilibrium iterations permitted;

d) Constraints -- incremental constraints can be modified between load increments;

e) Solution traces -- Simple traces of solution computations permit the analyst to determine if the analysis is proceeding satisfactorily, or if changes will be needed before the next load step.

4.6. Element and Material Model Libraries

During the lifetime of a finite element program, system functions, including input data translation, forming and solving the equilibrium equations, etc., remain relatively unchanged. Occasionally new features are added and errors corrected but the basic concepts of equation solving, data management, etc. do not appreciably change. In contrast, element formulations and models of nonlinear material behavior are still primary areas of current research and thus change quite frequently. The library of a general system will be constantly modified by users in a research environment. A measure of the versatility of finite element software is the relative ease with which users can install and test new elements and material models.

The following sections briefly summarize the function of element and material model libraries and mechanisms for interfacing library routines with the system.

4.6.1 Finite Element Libraries

An important feature of the finite element method is that all elements, regardless of their complexity, perform essentially the same function from a systems viewpoint. During nonlinear analysis, element dependencies are limited to the following specific functions:

a) Generate the tangent stiffness matrix consistent with the Lagrangian formulation given the element nodal displacements, stresses, coordinates, etc.

b) Generate an increment of Green strain given the previous total displacements and the increment of element nodal displacements.

In finite deformation problems, element strain routines must also generate a matrix of geometric transformation data that permits material models and output routines to transform between various definitions of strain and stress.

c) For linear or geometrically nonlinear elements, the element stress routine computes total stresses given the previous stresses and the Green strain increment. In materially nonlinear analysis, element stress routines typically perform no computations. Exceptions are some beam, plate, and shell elements, for example, that integrate stress components to form stress resultants.

d) Pressures, temperature gradients, and other applied element loads are converted to equivalent element nodal loads by element load routines. Effects of nonconservative loads, if present, are accounted for in these routines.

e) During residual load computation, element routines generate the internal nodal forces corresponding to the current element stresses and deformed geometry.

f) Some elements may generate special output data, such as principal stresses, or alter the number of points for output. This data is not incremental in nature and therefore cannot be computed during solution. These values must be computed by element routines just prior to output.

4.6.2 Material Model Libraries

Material models are necessary to idealize the behavior of real materials under complex states of stress in the nonlinear range. Incremental

deformation theories capable of modeling a broad range of behaviors are appropriate for general analysis. Model dependencies can be limited to the following operations:

a) Initialize history dependent quantities that describe subsequent material behavior. Form the elastic stress-strain relationship constitutive matrix.

b) Compute the new total stresses at a point within an element given the previous stresses, strain, history parameters, etc. and the increment of Green strain. Effects of creep, shrinkage, temperature, and time may be accounted for during these calculations.

c) Compute an instantaneous tangent modulus matrix for re-computing the element tangent stiffness matrix.

4.6.3 System-Library Interface

The main factor determining the ease with which users may modify the library of elements and material models is the interface between the system and the library routines. The fundamental requirement of the system-library interface is standardization. All interfaces should be identical regardless of the element or model complexity. Data made available to library routines should come directly through the system interface and not indirectly through FORTRAN COMMON areas from other library routines. The latter method leads to confusion for users not familiar with all other parts of the system. No elements or models should receive special consideration. For simple elements and models this leads to overkill in that more data than required is made available to the element and material model routines.

By standardizing the interface and forcing all data to pass through this interface, element and material models are isolated completely from the system. This permits the system to perform additional functions common to all elements and models. Memory allocation, translation of input data, printing of results, compatibility checking, etc. can all be handled by the system. Moving these tasks to the system level frees the developer to pursue his real interest -- performance testing of a new element or material model.

CHAPTER 5

IMPLEMENTATION OF FINITE ELEMENT SOFTWARE

5.1 General

Finite element software is now recognized by the engineering profession as a major link in the transfer of technology between researchers and practicing engineers [47,65,66]. Frequently, software represents the only practical result of a research project since, unlike the associated technical publications, it is already in a directly usable form.

Planning, design, and implementation of finite element software to achieve all the requirements outlined in the previous chapter constituted a major part of this research. Currently available nonlinear analysis programs achieve only a few of the requirements, primarily because they were developed by finite element researchers interested in the formulation of numerical algorithms, not in software engineering. As the size of structures increases, inadequacies of these programs become apparent, as does the need for careful planning and design of the software. The percentage of development effort devoted to numerical algorithms is relatively insignificant compared to that required for design of an internal system and data structure capable of efficiently processing the enormous volumes of structural data.

This chapter presents various design and implementation aspects of FINITE, a general system for linear and nonlinear structural analysis. FINITE represents a first attempt to synthesize both the analytical capability demanded by sophisticated analysts and modern software design technology into a comprehensive nonlinear analysis system.

The present chapter is divided into five sections for presentation. In the first, two major approaches of software development are delineated. The second section presents the fundamental concepts embodied in the POLO supervisor with examples illustrating their application to nonlinear analysis. A brief overview of FINITE, given in the third section, emphasizes the distinguishing characteristics of the system's internal organization. The fourth section considers nonlinear material modeling in FINITE from the user and new model developer points of view. The final section examines two of the primary nonlinear processing modules appended to the linear version of FINITE to produce the nonlinear version.

No attempt has been made here to fully describe details of the FINITE system (the current nonlinear version consists of more than 100,000 FORTRAN statements). Rather, the fundamental concepts of its design are shown to illustrate the primary differences between FINITE and other existing nonlinear finite element codes. Chapter 6 demonstrates the problem solving capability of FINITE through example analyses. Details of system usage can be found in the FINITE user's manual.

5.2 Structure of Finite Element Software

Interest in various approaches for developing finite element software arose as engineers began to examine the internal structure of computer programs and associated data. As a result of software development efforts over the past decade, two distinctly different approaches have emerged. In the first, only standard features of an algorithmic language, such as FORTRAN, are employed in the actual code. Such an approach places severe limitations on the types of program and data

structures available to developers. All logical operations on data must be repeatedly programmed by system developers. The vast majority of existing finite element software falls within this category. The second approach is distinguished by the presence of a comprehensive engineering data base and memory management system that relieves developers from programming these tasks. This approach has been termed "integrated" software in the literature. Characteristics of both approaches are discussed in the following sections.

5.2.1 Systems Without a Data Base Manager

Existing nonlinear finite element programs have a structure similar to that shown in Fig. 5.2.1. Problem data is divided among any number of sequentially accessed files each of which has a trivial structure. A large number of files is necessary (10 or more files is not uncommon) to represent data associated with a nonlinear analysis.

The computer code consists of FORTRAN subroutines grouped together into processing modules. One module contains the logic to direct execution of those at a lower level. Data is transmitted between modules in two ways. Simple control parameters are passed through standard features of the FORTRAN language. The majority of data is implicitly passed through the sequential files. Each module reads data into memory from appropriate files, performs the computations, then re-writes the data to the sequential file prior to initiating the subsequent module. Various schemes that maintain certain data in memory at all times have also been used. The SAP [7,71] family of codes employs a combination of sequential files and fixed data in memory.

Inadequacies of this approach become apparent when one tries to modify or expand the program capabilities, or to solve large problems. These programs become I/O bound (inordinate amounts of data transferred between memory and files) for large problems because each module performs its own programmed read/write operations irrespective of the memory space available. These programs cannot utilize additional memory if made available, and, because data is tied directly to physical locations in a sequential file, changes in the data structure usually necessitate re-writing much of the program. The definition of more files circumvents this difficulty at the expense of further complicating subsequent program modifications.

The simple data structures permitted on sequential files place severe limits on the level of sophistication achievable in these programs. Consider, for example, implementing a multi-level substructuring capability with sequential files. The many structural topologies, geometries, loads, computed results, etc. would require an excessive number of data files. With this approach to software development, implementation of such features is both unrealistic and impractical.

5.2.2 Systems With a Data Base Manager

A more realistic finite element system is illustrated in Fig. 5.2.2. A number of processing modules are developed that operate on a logical data space. A data management system, independent of all processing modules, resolves logical data requests issued by the modules into physical locations within a file. Thus, the mapping of the logical data space onto an actual secondary storage device is transparent to the processing modules. Each module may access any data within the data space. Interfacing between modules is accomplished easily through the logical data

structure. Ideally, modifications of the logical data structure as the system capability increases should not require changes in existing modules. A memory management system that interfaces with the data manager is necessary to map data from the physical file into memory for use by the processing modules.

A much simplified version of this ideal scheme has been implemented in a number of large linear analysis systems, the most notable of which are ASKA and NASTRAN. In these systems, references to data management routines are imbedded within the FORTRAN code of the individual processing modules. Most references simply extract a specified matrix from a file. Hierarchial data structures, required to implement many of the features described previously, are difficult to maintain in this manner. Engineers developing new features must write well planned sequences of references to data management routines to interact with a hierarchial data structure. Additional complications occur when defining and maintaining the logical description of the data space through references to special data management routines.

The POLO-FINITE system more closely approaches the system structure shown in Fig. 5.2.2. The POLO supervisor supports the logical data space concept and minimizes programming of data management tasks by providing developers with high level languages and compilers to do the work automatically. The FINITE system has a modular structure centered around a hierarchial, logical data space. This philosophy of design permits the system capability to evolve naturally without major revisions in existing modules. A combined interactive-batch processing capability with POL input makes the system flexible and well-suited for nonlinear analysis.

5.3 POLO -- An Engineering Software Support System

POLO was a prominant factor in the design and implementation of the nonlinear capabilities of the FINITE system. A brief description of POLO's capabilities pertaining to this study is presented here. Complete functional details of POLO may be found elsewhere [37].

POLO provides software engineers with an enhanced programming environment in which to develop engineering application software (POLO subsystems). POLO itself does not solve engineering problems; rather it supports functions common to all engineering application areas. These functions include:

1) Problem oriented language translation;

2) Data structure definition;

3) Data and memory management during execution;

4) Integration of application subsystems.

POLO supports two higher level languages, actually POLs, termed F and G. Subsystem developers describe the logical structure of data with the F language. These symbolic data definitions (File Definitions in POLO terminology) are converted to internal form and stored in a POLO data base independently of application subsystems. After this process is completed, subsystem developers write drivers for processing modules in the POLO host language G. Statements in the G language form "grammars" similar in function to a well structured FORTRAN main program. Once the grammar is written, the subsystem is completed by writing standard FORTRAN subprograms referred to by the grammar.

The primary purpose of a grammar is to direct execution of the supporting subprograms and to pass them data from a POLO data base, i.e.,

the grammar provides a logical interface between the data space and the FORTRAN subprograms. Data required by subprograms is referenced in the grammar using symbolic names provided in the file definition. Instructions for locating the data (data management requests) are generated automatically by POLO. The advantage of this approach is that subsystem developers need not be concerned with the physical mapping of data in the data bases. Grammars are also used to translate POLs and to initiate other POLO subsystems. A special POLO subsystem (compiler) converts the grammars to an internal form, termed object grammars, and saves them in a system data base.

During execution, the primary POLO interpreter executes the object form of the grammar. When instructed to invoke a subprogram referenced in the grammar, POLO routines make the requested data physically present in memory (if it is not already present), then pass the data to application subprograms as standard FORTRAN arguments. Data is made present in memory by a separate data management interpreter coupled with a demand paging dynamic memory allocator. Application subprograms are not aware of data and memory management functions as all operations of making data present are done prior to calling them.

In summary, POLO subsystems consist of three major components:

1) Symbolic file definitions,

2) Grammars,

3) Subsystem FORTRAN subprograms.

The following sections describe the first two of these components and illustrate particular applications to nonlinear analysis. Then a description of POLO's run-time configuration is presented.

5.3.1 File Definitions

Most data for finite element analysis is naturally structured in a hierarchial manner. Consider a simplified portion of the FINITE data structure shown in Fig. 5.3.1. The highest level table, called STRUCTURES, contains information about structures and individual finite elements. Each row describes an attribute for an element or structure in a particular column. Row 3 points to a lower level table, LOADS, that contains a description of the loads, in addition to results for the individual steps of a nonlinear analysis. Each column of the LOADS table contains results for a single load step. Load tables exist below only those columns of STRUCTURES that contain structures.

One set of results examined here for illustration is the geometric transformation matrices between deformed and undeformed element configurations for large displacement analysis. A matrix of geometric data is required for each strain point of every element that permits large displacements. Row EPSTRANSFORM of the LOADS table points to a lower level table containing pointers to vectors of transformation data for many elements. The vector number and position within the vector of the matrix for the first strain point of an element is sufficient to locate all matrices for the element (matrices are stored contiguously within a data vector). Row EPSVECPOS of the STRUCTURES table contains the vector number and position for the element's data.

The logical data structure described above is a combination of inverted lists and hierarchial tables. Similar constructs are used throughout FINITE to permit random access to nonlinear element data independently of the structure in which the elements are used. Somewhat

different data structures in FINITE permit rapid access to data for all elements of a specific structure.

The data structure described in this example demonstrates the fundamental kinds of data tables supported by POLO. STRUCTURES, LOADS, RESULTS, and EPSTRANSFORM are called labelled tables. Each row may contain different types of data (alpha, real, pointers, etc.). Such tables may consist of a single block of columns or multiple blocks of columns. New blocks of columns are made available on demand; however, grammar references are unaware of the internal blocking. The ETRNVEC table is the least flexible table type available. All data is of a single mode and the table is always contiguous in both memory and the data base.

Any number of hierarchies similar to the one shown in Fig. 5.3.1 can be defined within a POLO data base. The data structure shown in the figure is defined to POLO through the language F with the following commands.

```
FILE DEFINITION FINITE
 TABLE STRUCTURES LABELLED GROUPING 25
  SNAME ALPHA 2
  EPSVECPOS INTEGER
  LOADS LABELLED GROUPING 5
    NAME ALPHA 2
    RESULTS LABELLED 1
     DISPLACEMENTS LABELLED GROUPING 11
        .   .   .
     EPSTRANSFORM LABELLED GROUPING 11
      N INTEGER
      ETRNVEC VECTOR INTEGER N
     END OF TABLE
  END OF TABLE
END OF TABLE
  .
  .
  .
END OF FILE DEFINITION
```

5.3.2 Grammars

The primary function of a grammar is to initiate application subprograms passing them data from the data base. A single reference, no matter how deep in a hierarchy, is passed as one argument to the application subprogram. For example, in Fig. 5.3.1, assume the structure column number SCOL, loading column number LCOL, the vector number for element transformations VECNO, and position POS are all known. Data for the element can be passed to the application subprogram (with symbolic name TRANSFORM) via the following statement in the grammar

```
USE FINITE
    .
    .
    .
*EXECUTE TRANSFORM( STRUCTURES(LOADS,SCOL,RESULTS,LCOL,
                    EPSTRANSFORM,1,ETRNVEC,VECNO,POS) )
```

The USE command designates the file definition and appears only once in the grammar. The data reference above causes the POS entry of the data vector to be passed to the subprogram as an argument. The first few lines of the subprogram might be

```
SUBROUTINE TRNSFM( TVECTR )
DIMENSION TVECTR(1)
      .
      .
      .
RETURN
END
```

Traversals through the data hierarchy to resolve a logical data request are generally slow. Repeated references like the one above must be avoided to attain efficiency. POLO provides additional access mechanisms that considerably speed up execution. Artificial "base" tables can be defined at intermediate levels; subsequent lower references may then

begin at the intermediate level. Subsystem developers can manipulate the actual "pointers" and request POLO to perform direct accesses to data without the overhead associated with a traversal. The payoff is increased execution speed but with some increased programming effort.

5.3.3 Run-Time Configuration

During execution, POLO and the various FINITE subsystems appear as a single executable FORTRAN program as shown in Fig. 5.3.2. The POLO executive interpreter is the highest level driver in the system. It requests the token scanner to read POL commands from various input devices, convert free-form data to fixed-format data, and place the results in a FORTRAN COMMON area accessible by both POLO system routines and FINITE subsystem routines. The top portion of this vector is reserved for scanner output and a static area in which subsystem routines may place variables. The POLO dynamic memory allocator uses the remaining portion of the COMMON area to maintain requested data in memory.

As shown in the figure, application subprograms are totally isolated from the system functions necessary to locate data and make it present in memory. Data modified by the subprograms will eventually be forced into the data bases either during execution as the dynamic allocator seeks to make room for other data or at the end of execution when all data in the dynamic pool is written into the appropriate data bases. Thus, data is automatically maintained for future restarts.

5.4 FINITE System Organization

The logical organization of FINITE is best described by defining the three primary capabilities it provides structural analysts. These are:

1) Definition and maintenance of tables within a data base that describe the characteristics of element and material model library routines;
2) Definition and maintenance within a data base of geometric, topologic, load and constraint data for hierarchies of structures;
3) Computational and output procedures for analyzing structures and reporting the results.

To support the system capabilities outlined above requires a number of processing modules, data bases, and a mechanism to interface between the modules. The following sections briefly describe each of these components.

5.4.1 Organization of Processing Modules

At the most abstract level, the physical organization of FINITE consists of the following three components:

1) The POLO supervisory system;
2) FINITE processing modules that operate under control of POLO (POLO subsystems);
3) Element and material model subprograms initiated by FINITE subsystems.

An expanded diagram illustrating additional details of the system organization is shown in Fig. 5.4.1. POLO is represented as one functional module at the highest level with the primary FINITE subsystems immediately below.

Each FINITE processing module (currently 19 of them) is a separate POLO subsystem. The highest level processor, FINITE, ensures existence of appropriate data bases and translates POL commands to determine which of the three secondary subsystems to initiate (LIBRARY, STORE, or COMPUTE). Each of these subsystems corresponds directly to one aspect of the logical organization. The LIBRARY subsystem maintains descriptions of elements and material models supplied by their developers. STORE translates all POL statements defining structural geometries, loads, constraints, etc. COMPUTE contains the logic required to perform linear analysis by initiating lower level modules in the proper sequence. However, COMPUTE initiates NONCOMPUTE to direct solution of nonlinear structures. As shown in the figure, some modules, for example, the one for matrix decomposition, are invoked by several modules at different levels in the hierarchy. This illustrates the interdependency of processing modules but also shows that they are self-contained and can be easily initiated. Lower level subsystems invoke element and material model routines to compute stiffnesses, strains, stresses, etc.

5.4.2 Data Bases

The conceptual data structure for finite element analysis shown in Fig. 5.2.2 has been partitioned into three logical POLO data bases each of which resides on a separate physical file. The fourth file shown in Fig. 5.4.1 is the POLO system data base that contains object grammars and file definitions.

The data base denoted LIBRARY in the figure contains tables describing all finite elements and material models available in the system. Data in the tables are provided by developers of new elements and material

models through POL commands in the FINITE language. Geometric properties of common structural shapes can be defined in the library to minimize input data during structural definition.

The primary data base is denoted WORKFILE in the figure. The internal form of all structural descriptions and computed results reside in this data base. Extensive hierarchial data structures similar to that shown in Fig. 5.3.3 are defined to contain element and substructure geometry, topology, stiffness, loads, displacements, etc.

A separate data base, denoted SOLVER, is defined for solving equations. Triangulated equilibrium equations and results of condensations are retained in the SOLVER data base. These are available to process additional loading conditions as well as to retrieve displacements inside condensed substructures. The use of a data base separate from the WORKFILE permits optimal allocation of the equations on secondary storage to minimize I/O operations during equation solving.

5.4.3 Interfacing Between Subsystems

During analysis, various FINITE subsystems are invoked by other subsystems to perform parts of the solution process. Because of FINITE's flexibility, the exact order of subsystem invocation is highly problem dependent and is determined by the system as execution proceeds. Thus, a flexible technique for interfacing between subsystems was needed.

Conceptually, the process of invoking a hierarchy of subsystems is identical to a series of subprogram calls in FORTRAN. The basic actions required to initiate a new subsystem are:

1) Save the status of any variables in the currently executing subsystem to permit resumption of processing on return from the lower level subsystem;

2) Make available to the initiated subsystem controlling parameters passed by the initiating subsystem;

3) Maintain a hierarchial list of subsystems executed to reach the current level to enable re-tracing the order of execution.

The first two functions above are accomplished through a combination of global COMMON parameters and a stack of "request" vectors maintained as part of the logical data space. The data structure for this is shown in Fig. 5.4.2. The third function above is performed directly by the POLO supervisor.

To initiate a lower level module, the currently executing subsystem builds a request vector of parameters for initializing the subsequent module and pushes the request onto a stack. The global parameter REQLEVEL, available to all subsystems, indicates the current stack level. The active subsystem then requests POLO to initiate the lower level subsystem. Once executing, the new subsystem takes its instructions from the top of the stack.

This simple mechanism is used in all modern computers (hardware), and provided FINITE developers with an ideal scheme for maximum "appearance of integration" while maintaining in reality 19 different, and loosely interconnected subsystems.

5.5 Material Modeling in FINITE

The unique feature of nonlinear material modeling in FINITE is the isolation of elements and material models from the remainder of the system by standard interfaces. The procedure for defining new linear elements is described in Ref. [34]. Only minor changes were required to incorporate

additional data describing nonlinear elements. The discussion here presents the new aspects of FINITE to support a general nonlinear material modeling capability.

The interface mechanism adopted is identical for all material models regardless of their complexity. An important advantage of this approach is that developers of new material models need be concerned with the organization of only a small "subspace" of the entire FINITE system. Once an individual is familiar with the interface scheme, implementation of subsequent material models is straightforward.

In addition to isolating material models, FINITE performs several "systems" tasks that have inhibited model development efforts in previous nonlinear finite element systems. Translation of material model input data, display of results, memory allocation, data retrieval, and compatibility checking between elements and models are all handled by FINITE's material processing subsystems.

The following section describes how one uses existing material models in FINITE to analyze nonlinear structures. The procedures for implementing new material modeling capabilities are described in section 5.5.2.

5.5.1 Specification of Materials

During the input phase of analysis, nonlinear "materials" are created by the user through POL statements in the FINITE language. Finite elements are marked materially nonlinear by FINITE input translators whenever a nonlinear material is mentioned as part of an element description.

A "material" is defined by selecting three possible components from FINITE's library. These components are:

1) A material model;
2) A stress-strain function compatible with the material model (optional);
3) An initial-strain function compatible with the material model (optional).

The material model determines stresses in the material given the previous strain and stress, new strain, loading history, etc. Most classical models based on plasticity theory utilize a yield or loading surface with some flow rule. However, FINITE's material system is in no way restricted to plasticity type behavior. The stress-strain function provides the selected model with physical properties of the material, for example, results of a uniaxial tension test. The initial-strain function supplies the model with time dependent effects on material behavior.

The following POL statements define a simple material:

```
MATERIAL A36-STEEL TYPE VON-MISES
 PROPERTIES MAXINCR 30 ALPHA 0.05 DIVERGE 500
 USE STRESS-STRAIN FUNCTION SEGMENTAL
  PROPERTIES E 30000.  NU 0.3 SEGMENTAL,
             NUMPOINTS 3,
             STRAINS 0.0  0.001, 0.008,
             STRESSES 0.  36.  42.
```

The material name is A36-STEEL and its behavior is modeled by the VON-MISES material model selected from FINITE's library. Users may reference A36-STEEL when defining the properties of finite elements.

PROPERTIES are parameters defined for a model or function for which users can supply values during analysis. In the above, ALPHA, MAXINCR, and DIVERGE control computational algorithms within the VON-MISES

material model.

The stress-strain function SEGMENTAL selected from the library approximates the uniaxial tension, stress-strain curve by a sequence of straight lines. Data describing the curve are also given through a PROPERTIES statement following the function declaration.

The input data shown above is given prior to requesting a nonlinear analysis. Once the incremental-iterative solution process has begun, a user can modify values for model and function PROPERTIES whenever he has control between load steps. The existing material is referenced with the word ACCESS given prior to MATERIAL. Only those model or function properties that have changed must be re-entered. Different functions may also be associated with the model. This feature is most convenient for altering tolerances, model algorithms, etc. as the nonlinear solution progresses.

5.5.2 Entry of New Models and Functions

The definition of new material models and functions is divided into two parts -- tables and corresponding subroutines. Information stored in the tables is used by FINITE to make compatibility checks, translate input data and allocate memory prior to initiating the subroutines. The individual defining a new model or function enters both tables and subroutines. The table definition process is termed "soft" because the developer may alter the information at any time by simply re-entering the table data during a FINITE run. This is convenient for developers working with experimental material models that change frequently.

The POL statements defining the VON MISES table data are listed below:

```
DEFINE MATERIAL MODEL VON-MISES
 SUBROUTINE 1
 MATERIAL STRESS-STRAIN FUNCTIONS
     SEGMENTAL, RAMBERG-OSGOOD, ELASTO-PLASTIC
 HISTORY  5  WORDS
 ELEMENTS
  CSTRIANGLE,  L2DISOP, Q2DISOP, ......
 PROPERTIES
    PSTRAIN  LOGICAL FALSE
    TOLERANCE REAL  0.001
        .
        .
        .
   END OF PROPS
 SYMMETRY
END OF MODEL
```

The SUBROUTINE statement associates the tables generated via the above commands with FORTRAN subroutines that perform material model computations.

Names of functions and elements compatible with the material model supplied in the tables permit the FINITE input translators to check compatibility of the material model, stress-strain and initial-strain functions, and the element type during the input phase of analysis. By delaying all consistency checks until an analysis is performed, developers may define elements, models, and functions in the library in any desired order, i.e., reference to non-existent elements and functions is permitted.

The HISTORY statement declares the amount of working storage space to reserve for each strain point of each element that references the material model. Information stored in this space is made available to the model each time the strain point is processed during nonlinear analysis. Material models may use the space to retain load path dependent parameters governing material behavior etc. ... FINITE is unaware of the type or mode of data within the history space.

PROPERTIES permit material model developers to define keywords and default values for parameters associated with the model. Users may optionally override the default value declared in the table during problem solution as shown previously. Real, integer, logical, and alphanumeric modes of data are acceptable as properties. Properties may be either scalars (single-valued) or vectors (multi-valued).

Most material models generate a symmetric matrix relating incremental changes of stress and strain. The SYMMETRY statement declares this result. Models that compute a nonsymmetrical matrix, for example, those using non-associated flow rules, must declare this with a NONSYMMETRY statement. FINITE automatically marks elements associated with non-symmetric materials as having nonsymmetric stiffness matrices. Similarly, substructures in which these elements appear are also nonsymmetric as are the final equilibrium equations. FINITE handles this situation without user or developer intervention from input translation through solution of the nonsymmetric equations.

Tables defining stress-strain and initial-strain functions are simpler than those for a material model. The SEGMENTAL function referred to in the previous example is defined by the following:

```
DEFINE MATERIAL STRESS-STRAIN FUNCTION SEGMENTAL
  SUBROUTINE 1
  PROPERTIES
    YMODULUS  REAL  0.0
    NUMPOINTS  INTEGER 0
    STRAINS  VECTOR  REAL  0.0
    STRESSES VECTOR  REAL  0.0
      .
      .
      .
  END OF PROPERTIES
END OF FUNCTION
```

Complete table input data for several material models and functions are given in Appendix B.

5.5.3 Model and Function Subprograms

During problem solution, the FINITE material processor calls material model subprograms to perform computations. Based on the type of calculations required, the processor locates and makes present in memory all data needed by material models and functions. Data for a single strain point is extracted and passed to the material model for computation. This procedure continues until all strain points of materially nonlinear elements of the structure (and substructures) have been processed.

Names of material model routines are MTMXX where XX is the two-digit sequence containing the SUBROUTINE number and is taken from the library tables. The calling sequence is

```
SUBROUTINE MTMXX (PROPPT,PROPS,HISTRY,DMTRIX,NSTRN,OLDEPS,
                  OLIEPS,DIEPS,OLDSIG,NEWSIG,NWIEPS,GENDAT,
                  TRANS)
```

A summary of the most important parameters is given below. Those marked with an (*) are updated by the model depending on the type of computation requested by FINITE.

PROPS - is the vector of model property values.

HISTRY - (*) is the history data vector for the strain point.

DMTRIX - (*) is the tangent modulus matrix for the strain point currently incorporated in the element and structure stiffness matrix.

OLDEPS - is the vector of total strains for the previous load increment or iteration.

OLIEPS - is the vector of initial strains included in OLDEPS, i.e., OLDEPS-OLIEPS = total mechanical strain.

DEPS - is the vector of incremental strains for this load increment or iteration.

DIEPS - (*) is the vector of initial strains included in the incremental strain increment (DEPS). For a load increment (iteration = 1), this vector contains initial strains due to the real applied incremental load. Thereafter, it contains whatever was entered in the NWIEPS vector during the previous call to the material model, i.e., incremental-iterative creep, shrinkage, etc., strains.

OLDSIG - is the vector of total stresses for the previous load increment or iteration.

NWIEPS - (*) is a zeroed vector in which the model may place new incremental initial strains due to creep, shrinkage, etc., that occur during a load step or iteration. These initial strains are converted into equivalent forces by the element residual load routines.

GENDAT - (*) is a vector of general data some of which is updated by the model to indicate results of computation.

TRANS - a vector of geometric parameters generated by the element strain routine for use in transforming from one definition of strain-stress to another for geometrically nonlinear problems. Generally this vector contains data describing geometry of deformed elements.

Material model subprograms reference stress-strain or initial-strain functions via standard FINITE system interfaces termed MTMDSS and MTMDIS. Material models communicate with functions through a set of parameters termed a communication vector. The protocol for data in the vector is established by the particular function used. Most functions require and return similar data, thus vector protocols are nearly identical. To reference a function, the material model builds the communication vector and issues a call to MTMDSS or MTMDIS. These FINITE routines invoke the particular function subprogram passing the communication vector and function property values. Functions place computed results in the

communications vector after which successive returns are made to the material model.

It should be restated that the use of functions is not mandatory. Material models that require very simple material characteristics may include them in the model properties and thus eliminate the need for functions.

Names of function subroutines are MTSXX and MTIXX for stress-strain and initial-strain functions, respectively. XX is the subroutine number declared for the function in the library. The calling sequences are

```
SUBROUTINE MTIXX(COMVEC,PROPPT,PROPS,GENDAT)
```

and

```
SUBROUTINE MTSXX(COMVEC,PROPPT,PROPS,GENDAT)
```

The parameters are:

COMVEC - the communication vector constructed by the material model routine;

PROPPT - a vector of subscripts used to locate property values in PROPS;

PROPS - the vector of function property values;

GENDAT - a vector of general information passed to the function.

5.6 Nonlinear Processors of FINITE

After the input phase of analysis, users may request computation of nonlinear results for any number of consecutive load steps. Subsystem NONCOMPUTE contains the logic to perform the nonlinear solution given the user's request in a suitable internal form.

Fig. 5.6.1 is a simplified block diagram showing the major flow of control through NONCOMPUTE. A special data base initialization is

required if the request is for computation of load step number one. Another separate section of the NONCOMPUTE subsystem performs the detailed sequence of operations necessary to effect solution for a single load step.

After solution of all requested load steps, control is returned to the FINITE subsystem. If in an iteractive environment, the user may ask for output, modify solution parameters and continue the analysis, etc. For batch operation, the next commands in the input stream are processed. Eventually, data bases are secured in preparation for future analysis restart and execution is terminated.

The two major components of NONCOMPUTE, data structure initialization and details of a single step solution, are discussed in the following sections.

5.6.1 Data Structure Initialization

A convenient feature of FINITE is that nonlinear structural hierarchies are defined by users exactly as for linear analysis. In the linear version of FINITE, data structures naturally suited to take advantage of identical elements and substructures were developed. The stiffness matrix and equivalent nodal loads for a substructure are computed just once and are referred to by each occurrence of the substructure when used as an element in a higher level structure. This procedure is not valid for nonlinear substructures because they have unique stiffness matrices and loads for each occurrence in the hierarchy. An "expansion" process was designed and implemented that makes each occurrence of a nonlinear element or substructure a unique component in the structural hierarchy. Expansion is totally transparent to the user and does not destroy the

original linear data structure thereby permitting a linear analysis of the structure if desired at a later time.

Expansion of the structural hierarchy is performed during solution for the first load step. Important components of the process are outlined in Fig. 5.6.2. The order of operations is designed to eliminate wasteful computation of stiffness matrices for multiple occurrences of nonlinear elements and substructures which are always identical (linear) in the first load step. Once the linear stiffness, based on the unexpanded hierarchy, has been computed, nonlinear elements and substructures are duplicated in the hierarchy for each occurrence. Data independent of nonlinear effects, for example incidences and coordinates, are not duplicated; rather data for the original occurrence is referenced. After expansion of the hierarchy, additional data structures are initialized for nonlinear data. Space is created for the incremental constitutive matrices and history parameters for all materially nonlinear elements. For geometrically nonlinear elements, a data structure is initialized to contain element local displacements for rapid access independent of the structure in which the elements reside.

5.6.2 Solution for a Load Step

The order of computations for a single load step is shown in Fig. 5.6.3. At the NONCOMPUTE level, the solution process is deceptively simple. The load step driver of NONCOMPUTE invokes appropriate lower level subsystems in the proper sequence corresponding to the solution parameters declared by the user. Complexities due to multi-level substructuring and condensation are handled by the individual lower level subsystems.

The first iteration of a load step corresponds to application of the real load increment. Subsequent iterations, if permitted and required, are performed at a constant real load level to remove the residual nodal loads and to bring the structure into an equilibrium configuration.

An important aspect of the solution process as outlined in the fiugre is data security in the event of analysis failure. If the solution for a load step fails to converge, the user can redefine the load step and request a new solution. To permit this, the system retains enough data between load steps to reconstruct the solution status at any point. Limits on the amount of secondary storage available prevent saving of previous tangent stiffness matrices and the triangulated equilibrium equations for each step. Only data necessary to reconstruct element and structure stiffnesses are saved. For example, if material nonlinearities are present, the incremental constitutive matrices and history parameters are retained. Thus, if a stiffness regeneration is required before processing can resume, the user simply indicates this through a solution parameter then requests solution for the same step again. Nonlinear FINITE subsystems automatically destroy and recompute data rendered invalid as a result of the re-analysis.

For large structures or structures analyzed with many load steps, the amount of data on secondary storage may become very large. Commands are available for users to indicate that results for specific load steps need no longer be retained. Space on secondary storage occupied by this data can then be reused.

In the current nonlinear version, once computations begin for a load step the user cannot regain control over execution until the step computations are completed. In the future, provisions should be made to permit resumption of load step computations in the event of nonconvergence.

CHAPTER 6

NUMERICAL EXAMPLES

6.1 General

The nonlinear capabilities in the FINITE system, as designed and implemented in this study, are demonstrated through a series of example analyses in this chapter. A variety of structural configurations and nonlinear effects are considered. Realistic structures in both size and complexity were chosen for examples to illustrate the practical aspects of the system.

These problems are not intended to examine the adequacy of any particular finite element or material model formulations. They demonstrate that a) material models can be easily implemented, b) FINITE can handle a variety of nonlinear behaviors, and c) and the system is relatively easy to use. Elements and models employed in the examples have already proven effective in other studies. Emphasis here is placed on structural modeling, data generation, ease of input, multiple analysis restarts and other aspects of the system experienced directly by users. Of particular interest is the application of substructuring and condensation to reduce analysis cost.

The first two examples illustrate the analysis of structures composed of long slender members in which nonlinear behavior occurs as a result of large rigid body rotations. A space truss dome and plane frame building are the two structures considered. In both problems the incremental solution method provides deformations prior to buckling as well as the approximate critical load for the structure.

The last three examples consider the effects of nonlinear material behavior in solid mechanics problems. Plastic deformation near a junction in a steel axisymmetric pressure vessel using the Von Mises criterion is examined in the first solid mechanics problem. The second problem simulates a tunnel opening in rock using the Drucker-Prager yield criterion. The last example is a three dimensional plastic analysis of a perforated thick plate subjected to uniform pressure.

6.2 Space Truss

Reticulated domes constructed of truss-type members have long been an economical solution for roof structures. Most analyses in the past were based on linear theory even though shallow domes were known to respond nonlinearly under design loads due to changes in geometry. Under higher load levels, shallow domes exhibit snap-through buckling behavior. Even more important from a design consideration is the sensitivity of these structures to nonsymmetric loads resulting from wind pressure and snow.

One particularly interesting geometric configuration is the Schwedler dome shown in Fig. 6.2.1. There are 24 equal divisions on each latitudinal circle and 4 equal divisions along each meridian. All nodes of the dome lie on the surface of a sphere. Base nodes are completely restrained against translation. Dimensions and material properties are shown on the figure.

The structure has cyclic symmetry but does not have radial symmetry; thus all nodes must be included in any analysis. Under a uniform vertical load the structure twists counter-clockwise about the vertical axis.

The finite element formulation for a general axial force member including all nonlinear strain terms is given in Appendix A. The significance

of retaining all nonlinear terms has been demonstrated in Ref. [30] which also solved the Schwedler dome structure. Omission of certain terms leads to straining of an element subjected to a rigid body rotation.

The analysis here considers two separate loading conditions. First, a uniform gravity load of 60 psf is applied over the entire dome. The second consists of a uniform vertical load acting on one half of the dome designated ABC in the figure. The magnitude of this load is increased in a series of increments until instability occurs. For both loading conditions, equivalent nodal loads corresponding to an applied vertical load of 1 psf form the basis of nonlinear load increments.

Two structural models were developed for analysis. The first model includes all nodes and elements as shown in Fig. 6.2.1 and is termed the standard model. An examination of the structure indicates that nonlinear effects should be significant in the nearly flat part of the dome shown in Fig. 6.2.2c. Steeper meridians in the lower portion are not significantly influenced by small geometry changes. The lower portion is modeled as a linear substructure with condensation applied to eliminate all nodes except those connecting the top ring. The substructured model is shown in Fig. 6.2.2.

The incremental-iterative analysis procedure was carried out using two load steps for the full gravity loading. Five load steps were used to attain the instability load for the partial loading. Iterative corrections for residual loads were performed at each step. Convergence checks were based on the residual load vector norm. A rapidly increasing residual load norm indicates the instability load has been reached. At the critical load level, the structure attempts to snap-through to the

alternate equilibrium configuration Since this configuration is of no practical interest, the analysis was terminated.

Portions of the FINITE input data for the two structural models are listed in Fig. 6.2.3 and 6.2.4. Both sets of input data illustrate the problem-oriented-language commands to describe a structure, its loads, and the analysis procedure. Input data for both models is straightforward and self-documenting.

Results for the uniform load case are shown in Fig. 6.2.5. The numerical values given are for the standard model. Deflections for the substructured model (not shown) are slightly larger as the stiffening effect due to geometry changes is not fully accounted for. However, the differences are less than 3%. Results shown in Fig. 6.2.6 for the partial loading condition are for the load level (29 psf) just prior to instability. The standard model did not converge at a load level of 29.5 psf. The substructured model did not converge at a load level of 30 psf. Little difference is noted between the standard and substructured models thus verifying the original assumption regarding the location of nonlinear effects. The primary difference between the standard and substructured models is the computer time required for analysis. For the full uniform loading condition, the substructured model required 74% as much time as the standard model (2 load steps). More noticeable savings are achieved for the partial loading condition. For that case, the substructured model used only 53% as much time as the standard model (5 load steps). Even greater savings would occur with a more detailed analysis involving a larger number of steps.

6.3 Second Order Frame Analysis

Linear analysis of plane frame structures neglects the interaction of axial forces and bending stiffness. This interaction becomes significant in braced frames with large column loads and, in all unbraced frames. Current design codes require amplification of member forces computed by a linear analysis to account for interaction effects. Magnification factors applied to linear results have been developed from approximate nonlinear analyses. For practical design, stresses remain well below the proportional limit; thus, the determination of axial load effects on bending action involves only geometrical nonlinearity.

Most design codes permit a more accurate determination of member forces in frame structures by a "second order" analysis procedure. A second order analysis must rationally determine the effect of axial loads on rotational stiffness coefficients in braced frames and the P-delta effect in unbraced frames.

Large compressive column loads in braced frames reduce the moment exerted by the column on any rigidly framed beams, thereby increasing deflections and positive beam moments. Buckling of a braced frame occurs when the applied load drastically reduces column rotational stiffness coefficients.

Vertical loads acting through sway deflections of unbraced frames produce additional bending moments and shears. This phenomena is termed the P-delta effect. Member forces in unbraced frames determined by a second order analysis include this effect and are not modified by the amplification factors in design codes. Unbraced frame members are proportioned to resist second order forces using the same procedures for braced frames.

The nonlinear plane frame element in FINITE has three degrees of freedom (U, V, θ_z) at each node as shown in Fig. 6.3.1. Element stiffness characteristics are determined from the classical differential equation of the elastic curve including axial load

$$E\ I\ V''(X) = (\pm)\ P\ V(X) \qquad (6.3.1)$$

where E is Young's modulus, I the moment of inertia in the plane of bending, P the axial force, and V(x) is the lateral deflection along the member measured from the undeformed configuration. Tensile and compressive axial forces are permitted. The element secant stiffness is derived in Refs. [9, 39] using stability functions obtained from homogeneous solutions of this equation.

Approximations employed in the development of the differential equation must be carefully delineated for practical nonlinear analysis. They are:

1) The curvature of the member is infinitesimal and may be adequately described by the second derivative of lateral displacement, V"(X). This follows directly by assuming the member centerline slope squared, [V'(X)], is negligible compared to 1.
2) Axial forces in members may be determined by the equation

$$P = E\ A\ e/L \qquad (6.3.2)$$

where e is the member elongation given by $U_4 - U_1$ as shown in Fig. 6.3.1. A is the cross-sectional area and L is the undeformed length. This assumption implies infinitesimal member length changes and that changes in direction of member end forces due to the sway angle are neglected.

3) The deflected shape of a laterally loaded member subjected to an axial force is adequately described by equivalent forces applied at the ends as determined from the deflected shape without axial force.

These assumptions have no significant consequence in practical analysis and design. Drift limitations imposed by building codes render 1) and 2) valid. Normally beams are the only members subjected to lateral loads and the effect of small axial forces is very insignificant; thus, the third approximation is valid.

The nonlinear plane frame element in FINITE adequately predicts the second order effects as required by design codes for use as an alternate analysis method. The element predicts classical linear buckling loads for braced frames and for unbraced frames if the pre-buckling deformations are small. The following two examples solved with FINITE illustrate simple, but practical applications of the nonlinear capability.

P-Delta Analysis

An unbraced plane frame structure with dimensions and loads shown in Fig. 6.3.2 was analyzed for P-delta effects. Deflections from the ficticious lateral load method described by Adams [1,2] are compared with FINITE results. Following Adams, the column loads for P-delta analysis are increased by 1.7 over working loads to approximate ultimate conditions. Although some researchers disagree with this philosophy, the increased loads are used here to permit direct comparison with the published results of Adams. Fig. 6.3.3 summarizes the lateral displacements computed by a standard linear analysis, Adam's method, and FINITE.

Displacements computed by FINITE are larger than those predicted by the lateral force method. This follows because the shape functions used in the FINITE formulation are exact within classical theory. The lateral force method tries to correct for unbalanced element end forces using the linear stiffness matrix. Since both models are "compatible" in finite element terminology, the one based on correct shape functions is less stiff.

Input data required to analyze the structure with FINITE is listed in Fig. 6.3.4. All beams were assumed to respond linearly. Two load steps were used; each step required 3 iterations for convergence with stiffness updates performed before the second iteration of each step.

Linear Buckling Analysis

The nonlinear plane frame element in FINITE can be used to compute approximate buckling loads as illustrated in this example. The plane frame structure shown in Fig. 6.3.5 is subjected to a distribution of vertical loads corresponding to self weight, superimposed dead load, and uniform live load on all girders. Very small lateral loads insure the frame will buckle in a sidesway mode. The initial loading pattern represents the magnitudes for a unit value of the loading parameter λ. During analysis, proportional loads are applied by simply incrementing λ. Instability of the frame occurs when lateral displacements increase rapidly for small changes in λ. The critical load level is denoted λ_{cr}.

Estimates of λ_{cr} based on displacements for value of $\lambda < \lambda_{cr}$ are computed from

$$\lambda_{cr} \cong \frac{\lambda \, \Delta_{n\ell}}{\Delta_{n\ell} - \Delta_{\ell}} \qquad (6.3.3)$$

where Δ_{ℓ} and $\Delta_{n\ell}$ are the linear and nonlinear displacements of any story corresponding to loading level λ. A derivation and discussion of this useful formula are given in Ref. [39]. An average λ_{cr} computed from λ_{cr} of each story provides a guideline for the next increment of loading to apply.

Fig. 6.3.6 presents a summary of the analysis results. The P-delta load deflection curve for the top story is given in Fig. 6.3.7.

6.4 Axisymmetric Pressure Vessel

A steel axisymmetric pressure vessel with a cylinder to sphere junction is shown in Fig. 6.4.1. Design of these vessels normally follows elastic theory supplemented by standard design codes of practice. Stress distributions are often determined from simple membrane theory with stress concentration factors to account for bending and localized effects. The concentration factors are generally the result of elastic thin shell solutions; however, shell theories are unable to easily account for the stiffening effect of the junction weld. Upper and lower bounds for the critical pressure can be determined by approximate limit analyses.

The finite element method is naturally suited for analysis of this type structure. Nonlinear behavior is isolated within a small distance of the junction. This example illustrates 1) the specification of materially nonlinear problems in FINITE and 2) substructuring and condensation in stress concentration problems.

A vessel tested by Dinno-Gill [16] was chosen for analysis to permit comparison of the finite element solution with experimental results. Geometry of the vessel is shown in Fig. 6.4.1. Failure occurs well below the level of loading to cause large deformations; thus only material

nonlinearity is considered. The vessel was constructed of mild steel and stress relieved after completion of all welding. The Von Mises yield criterion adequately predicts material behavior under these conditions.

Eight node isoparametric elements were chosen to idealize the vessel. Strains vary linearly over each element. Nonlinear elements are used only in the immediate vicinity of the cylinder-sphere junction. Remaining elements in the sphere and cylinder are assumed to respond linearly for all load levels. The finite element mesh is shown in Fig. 6.4.2. Reduced integration, 2 x 2 rule, employed in the linear regions reduces cost and improves element performance. In the nonlinear elements, full 3 x 3 integration permits early detection of yielding due to high bending stresses near the inner and outer vessel wall surfaces.

The multi-level substructuring features of FINITE employed in this problem reduce input data and analysis costs. The linear portion of the cylinder is modeled with two levels of substructuring to take advantage of identical elements in the cylinder wall. Static condensation applied to both the linear sphere and cylinder substructures eliminates all but three nodes necessary to connect with nonlinear elements as shown in the figure. The final structure consists of the 36 nonlinear elements and the two condensed substructures. A unit pressure load applied to the inside face of all elements in the linear substructures and nonlinear elements constitutes the loading condition. Scalar multiples of this loading define the individual step load increments. Other aspects of the analysis are described in the input data shown in Fig. 6.4.3.

The spread of plastic zones for the levels of internal pressure are shown in Fig. 6.4.4. The load deflection curve for node 59, shown in Fig. 6.4.5, agrees well with experimental results.

Several aspects of the solution process from the user viewpoint are worthwhile considering. Data generation commands simplify the input of nodal coordinates and element topology. Loadings on lower level substructures are carried upward through the hierarchy via the EXTERNAL LOADS command. The nonlinear processor of FINITE recognizes that the substructures are linear and requests computation of the condensed stiffness matrix and equivalent loads only once during the entire solution. A nonlinear MATERIAL termed STEEL is defined for association with finite elements describing the vessel. The material model (yield function, flow rule, and hardening rule) selected from the FINITE material library is called VON-MISES. Details of this model are given in Appendix B. The properties TOLERANCE, ALPHA, and MAXINCR, specified by the user, unless default values are adequate, control numerical algorithms within the material model. Physical characteristics of the material are specified by a stress-strain function selected from the library (SEGMENTAL in this problem). This particular function permits any type of segmental stress-strain curve to be input with a shortened form available for elasto-plastic behavior as in this example. Elements in the junction structure are declared materially nonlinear simply by specifying MATERIAL STEEL after stating the element type.

6.5 Deep Tunnel in Rock

A simple application of the finite element method to a nonlinear geotechnical problem involves excavation of a tunnel in deep rock.

Stresses around tunnels in deep rock were previously investigated with the finite element method by Reyes and Deere [60].

The particular problem chosen for presentation here is a 20 ft. diameter tunnel excavated in rock. The Mohr-Coulomb yield surface is approximated by the analytically simpler Drucker-Prager yield criterion for plane strain as discussed in Appendix B. This material model adequately represents the response of rock for small strains.

The infinite medium surrounding the tunnel is represented by a finite element mesh extending five tunnel diameters away from the tunnel face as shown in Fig. 6.5.1b. Disturbances due to the tunnel are small at this distance. Twenty-eight, eight-node quadratic isoparametric elements were chosen to simulate the region near the tunnel. Ref. [60] employed 480 constant strain triangles. Double symmetry planes permit use of only one quarter of the continuum.

Loads applied to the model simulate the construction sequence. In situ stresses of 1000 psi vertically and 250 psi horizontally are introduced by appropriate loads applied along the exterior boundaries and the tunnel face. Under these stresses, no elements are yielded. Incremental loads applied to the tunnel face gradually reduce the tractions to zero (representing completion of the excavation).

Convergence was rapid with the tangent stiffness matrix updated before the second iteration of each step. Most of the FINITE input data is given in Fig. 6.5.2.

The spread of plastic zones is indicated in Fig. 6.5.1a. The redistribution of stresses due to yielding is shown in Fig. 6.5.3. For both this and the previous study, yielding was found to spread at a

forty-five degree angle away from the tunnel face. Differences in the plastic zones occurring in this study and those of Ref. [60] are attributed to the different element meshes. However, the differences are negligible for practical purposes.

6.6 Thick Penetrated Plate

This final example illustrates the solution of a large (by current standards) nonlinear three dimensional structure. The multiple restart feature of the FINITE system was employed to partition the total solution process into manageable segments.

A 180^{o} section of a thick circular aluminum plate is shown in Fig. 6.6.1. The plate has a diameter of 14 in. and is 1.25 in. thick. Three penetrations with chamfered edges are placed 90^{o} apart. The Von Mises material model with an elastic-plastic uniaxial material stress-strain curve was selected to represent material behavior. A total of 80, 20-node isoparametric solid elements model the plate. Twenty-eight elements on the plate periphery were declared linear to reduce computation time. Standard 3 x 3 x 3 Gaussian integration within these elements is adequate. A numerical integration order of 3 x 3 x 5 (5 layers of 9 points) in the nonlinear elements permits early detection of yielding through the depth caused by large bending stresses. The structure has 665 nodes with a total of 1995 degrees of freedom. Nodes along the outside edge of the lower surface were completely restrained against translation. Input data is summarized in Fig. 6.6.2.

The plate was analyzed for a 100 psi uniform pressure from which the pressure at first yield was computed as 760 psi. Three load increments were then applied corresponding to total pressures of 800, 1500, and

2500 psi. The elastic stiffness was used for the 100, 800, and 1500 psi load levels. A new tangent stiffness was generated before the second iteration of step 4 (2500 psi).

The multiple restart feature of FINITE was utilized to solve each load step in a separate computer run. After the solution for a step converged, the displacements and stresses were printed to determine the load-deformation response and the extent of yielding. The analysis was restarted via the commands shown in Fig. 6.6.3 for each subsequent step. The first command requests POLO to execute the FINITE subsystem with data bases containing previously computed results for the plate. The plate structure is accessed and a new load increment defined. Convergence tests, tolerances, output commands, etc. can also be given. Input data shown in the figure also illustrates the DESTROY command that allows the analyst to release space on secondary storage containing results no longer required. Released space is re-used by FINITE to eliminate unnecessary expansion of secondary storage requirements. The final commands request computation of displacements and output of results for a load step.

Segmentation of the total analysis into multiple computer runs is mandatory for large structures. Even on large scientific computers the exposure time (elapsed wall clock time) can become excessive in multi-processing environments. The plate structure solution discussed in this example required over 28 hours of elapsed time on a Burrough's B-6700 computer. CPU times were 272, 35, 173, and 374 minutes for steps 1 to 4 respectively.

The spread of plastic zones is indicated for the various load levels in Fig. 6.6.4. Yielding begins between the two cutouts due to large

bending stresses then quickly spreads over the top and bottom surface before penetrating through the depth. Similar behavior is observed in plastic analysis of beams. Extensive yielding at 2500 psi applied load indicates the useful capacity of the plate has been exceeded.

CHAPTER 7

SUMMARY AND CONCLUSIONS

7.1 Summary

This study examined the principles of nonlinear finite element analysis and incorporated them with the principles of modern software engineering to produce a readily usable tool. Equal emphasis in the presentation was placed on the theoretical formulation and the software engineering aspects of finite element analysis.

The governing equations of nonlinear continuum mechanics, usually expressed in tensor notation, are presented in matrix form suitable for direct application in the finite element method. Both geometric and material nonlinearities are considered. The Lagrangian formulation was chosen over the Eulerian formulation for reasons of computational efficiency and convenience for the analyst. Strain-displacement relations and stress definitions appropriate for the Lagrangian description are presented and physically interpreted. Geometric relationships derived between the initial and deformed structural configuration enable the analyst to transform between stress definitions as convenient during analysis.

Classical equations of the Lagrangian formulation are cast into the finite element method. Specific matrices given for large deformation of a three dimensional continuum illustrate the procedure. Solution of the resulting nonlinear algebraic equations is accomplished with the Newton-Raphson method and its various modifications. A technique to account for nonconservative loads is discussed. Substructuring and static condensation

are proposed to reduce the cost of nonlinear analysis by decreasing input preparation effort and by eliminating consideration of linear regions of the structure during nonlinear computations.

The characteristics and features required of a general nonlinear finite element computer system from the analyst's viewpoint are examined. These include techniques for user-system communication, description of a nonlinear structural model via substructuring, analysis restart, and error recovery. Characteristics of an interface mechanism between the system and element-material model libraries are described that minimize the effort to implement new features.

An implementation of these features within the FINITE structural mechanics system is described. The role of the POLO supervisor in supporting problem-oriented-language translation, data management, and dynamic memory allocation are briefly examined. Examples of hierarchal data structures for representing data associated with nonlinear analysis illustrate typical constructs employed in FINITE. Details of the system-library interface scheme designed during this study are presented to show that finite elements and material models can be isolated from the system and each other thereby producing easily modifiable software. The overall processes of nonlinear analysis including data structure initialization and solution for a single step are described.

A number of nonlinear example analyses demonstrate features of the FINITE system. User-defined substructuring, condensation, and sophisticated data generators reduce analysis effort and cost. Both geometric and material nonlinearities are considered in the examples.

7.2 Conclusions

The basic equations of nonlinear continuum mechanics can be expressed in matrix form familiar to most structural engineers and can be readily incorporated within the finite element method.

Proper design and implementation of computer software for finite element analysis has now become recognized as an essential component in the overall analysis process. User-oriented, general purpose software makes the latest technological advances readily available to the engineering profession.

Engineering software support systems, commonly referred as supervisors, are a necessary and natural component of sophisticated structural analysis software. The use of a supervisor centralizes data and memory management functions thereby reducing development time and cost. In addition, a supervisor provides a convenient environment in which developers may incorporate good software engineering principles.

A comprehensive, multi-level substructuring and static condensation capability makes feasible the nonlinear analysis of many structures heretofore considered impossible or economically infeasible. The full potential of these techniques are not realized unless multiple levels of substructuring with condensation, including loads, is supported automatically by the software.

The useful life span of a nonlinear finite element system depends primarily on the flexibility designed into the interface between the system and element-material model libraries. A significant portion of the effort required to incorporate new elements and models can be removed from the developer and placed at the system level.

LIST OF REFERENCES

1. Adams, P., Beaulieu, D., Wood, B. R., "Column Design by the P-Delta Methods," J. of the Structural Division, ASCE, Vol. 102, No. ST2, Feb. 1976.

2. Adams, P., Nixon, D. and Beaulieu, D., "Simplified Second Order Frame Analysis," Canadian Journal of Civil Engineering, Vol. 2, No. 602, 1975.

3. Alwood, R. and Maxwell, T., "GENESYS--A Machine Independent System," CEPOC, University of Liege, Belgium, 1972.

4. Araldsen, P., "An Example of Large-Scale Structural Analysis of an Oil Tanker," J. of Computers and Structures, Vol. 4, 48, No. 1, 1974.

5. Balmer, H., Doltsinis, J., and Konig, M., "Elastoplastic and Creep Analysis with the ASKA Program System," Computer Methods in Applied Mechanics and Engineering, No. 3, 1974.

6. Baron, F. and Venkatesan, M., "Nonlinear Formualtions of Effects," J. of the Structural Division, ASCE, Vol. 97, No. ST4, April, 1971.

7. Bathe, K-J. and Wilson, E. L., "NONSAP--A General Finite Element Program for Nonlinear Dynamic Analysis of Complex Structures," Proceedings, Second International Conference on SMIRT, Berlin, 1973.

8. Bathe, K-J., Wilson, E. L., and Peterson, F. E., "SAP-IV--A Structural Analysis Program for Static and Dynamic Response of Linear Systems," EERC Report No. 23-11, University of California, Berkeley, 1973.

9. Beaufait, F., et al., Computer Methods of Structural Analysis, Prentice-Hall, Inc., New Jersey, 1973.

10. Bell, Hatlestad, Hansteen and Araldsen, NORSAM User's Manual, Trondheim, Norway, 1973.

11. Bushnell, D., "Large Deflection Elastic-Plastic Creep Analysis of Axisymmetric Shells," ASME, 1973 Annual Winter Meeting, Detroit, Michigan, November 1973.

12. Chen, H. C. and Schnobrich, W. C., "Nonlinear Analysis of Intersecting Cylinders by the Finite Element Method," Civil Engineering Studies, SRS No. 435, University of Illinois at Urbana-Champaign, December, 1976.

13. Cook, R. D., Concepts and Applications of Finite Element Analysis, John Wiley & Sons, New York, 1974.

14. Cook, R. D., "More on Reduced Integration and Isoparametric Elements International J. of Numerical Methods Engineering, Vol. 3, pp. 275-290, 1971.

15. Desai, C. S. and Abel, J. F., Introduction to the Finite Element Method, Van Nostrand Reinhold Company, New York, 1972.

16. Dinno, K. S. and Gill, S. S., "An Experimental Investigation into the Plastic Behavior of Flush Nozzles in Spherical Pressure Vessels," International J. of Mechanical Science, Vol. 7, pp. 817-883, 1965.

17. Dodds, R. H., and Lopez, L. A., "Experiences with Substructuring and Static Condensation," Proceedings, ASCE Conference on Computers in Civil Engineering, Atlanta, Ga., June 1978.

18. Drucker, D. C. and Prager, W., "Soil Mechanics and Plastic Analysis Limit Design," Quarterly J. of Applied Mathematics, Vol. 10, pp. 157-165, 1952.

19. Egeland, O. and Araldsen, P., "SESAM-69, A General Purpose Finite Element Method Program," International J. of Computers and Structures, Vol. 4, No. 1, January 1974.

20. Ergatoudis, J. G., Irons, B. and Zienkiewicz, O.C., "Curved, Isoparametric, Quadrilateral Elements for Finite Element Analysis," International J. of Solids and Structures, Vol. 4, pp. 31-42, 1968.

21. Felippa, C. A. and Scharifi, P., "Computer Implementation of Nonlinear Finite Element Analysis," Proceedings, Winter Annual Meeting of the ASME, Detroit, Michigan, 1973.

22. Fenves, S. J., Logcher, R. D. and Mauch, S. P., "STRESS--A User's Manual," MIT Press, 1964.

23. Frey, F. and Cescotto, S., "Some New Aspects of the Incremental Total Lagrangian Description in Nonlinear Analysis," Formulations and Computational Algorithms in Finite Element Analysis: U.S.-German Symposium, MIT, August 1976.

24. Fung, Y. C., Foundations of Solid Mechanics, Prentice Hall, Inc., 1965.

25. Gill, S. S., The Stress Analysis of Pressure Vessels and Pressure Vessel Components, Pergamon Press, Oxford, 1970.

26. Green, A. E. and Adkins, J.E., Large Elastic Deformations and Nonlinear Continum Mechanics, Carendon Press, Oxford, 1960.

27. Gupta, A. K., Mohraz, B., and Schnobrich, W. C., "Elasto-Plastic Analysis of Three-Dimensional Structures Using the Isoparametric Element," Civil Engineering Studies, SRS No. 381, University of Illinois at Urbana-Champaign, 1971.

28. Hand, F. R., Pecknold, D. A. and Schnobrich, W. C., "A Layered Finite Element Nonlinear Analysis of Reinforced Concrete Plates and Shells," Civil Engineering Studies, SRS No. 389, University of Illinois at Urbana-Champaign, August 1972.

29. Hatfield, F. J. and Fenves, S. J., "The Information Organizer: A System for Symbolic Manipulation," J. of Computers and Structures, Vol. 1, 1971.

30. Jagannathan, D., Epstein, H. and Christiano, P., "Nonlinear Analysis of Reticulated Space Trusses," J. of the Structural Division, ASCE, Vol. 101, No. ST12, December 1975.

31. Logcher, R. D., et al., ICES STRUDL-I, The Structural Design Language, Dept. of Civil Engineering, MIT, Cambridge, 1967.

32. Lopez, L. A., "FILES--Automated Engineering Data Management," J. of the Structural Division, ASCE, April, 1975.

33. Lopez, L. A., "POLO--A System for Integrated Systems Development," Proceedings, International Symposium on Integrated Civil Engineering Systems, CEPOL, Liege, Belgium, 1972.

34. Lopez, L. A., Dodds, R. H., Rehak, D. R. and Urzua, J.,"FINITE--A POLO II Subsystem for Structural Mechanics," Civil Engineering Systems Laboratory, University of Illinois at Urbana-Champaign, 1976.

35. Lopez, L. A., "FINITE: An Approach to Structural Mechanics Systems," International J. of Numerical Methods in Engineering, Vol. 11, No. 5, 1977.

36. Lopez, L. A., Dodds, R. H., Rehak, D. R., and Urzua, J., "Data Management Applied to Structural Problems," Proceedings, ASCE Conference on Computers in Civil Engineering, Atlanta, Ga., June 1978.

37. Lopez, L. A., "POLO--Problem Oriented Language Organizer," International J. Computers and Structures, Vol. 2, No. 4, 1974.

38. MacNeal, R. and McCormick, C. W., "The NASTRAN Computer Program for Structural Analysis" International J. Computers and Structures, Vol. 1, No. 3, 1971.

39. Majid, K.E., Nonlinear Structures, Wiley, New York, 1972.

40. Mallett, R. H. and Marcal, P.V., "Finite Element Analysis of Nonlinear Structures," J. of the Structural Division, ASCE, Vol. 94, No. ST9, pp. 2081-2105, 1968.

41. "MARC-CDC Nonlinear Finite Element Analysis Program," User Information Manual, Vol. 1, Control Data Corporation, 1971.

42. Meijerz, P., "Review of the ASKA Programme," Proceedings, NONR Conference on Structural Mechanics, University of Illinois at Urbana-Champaign, 1971.

43. Mendelson, A., Plasticity--Theory and Application, MacMillan, New York, 1968.

44. Mondkar, D. P. and Powell, G. H., "Static and Dynamic Analysis of Nonlinear Structures," UC EERC Report No. 25-10, University of California, Berkeley, March, 1975.

45. Murnaghan, F. D., Finite Deformations of an Elastic Solid, John Wiley & Sons, 1951.

46. Murphree, E. L., and Fenves, S. J., "A Generator for Translators of Certain Problem Oriented Languages," Civil Engineering Studies No. 1, University of Illinois at Urbana-Champaign, September 1967.

47. Nagy, D. A., "Software Engineering for Structural Analysis," Proceedings, ASCE Fall Convention, San Francisco, 1977.

48. Nayak, G. C. and Zienkiewicz, O.C., "Convenient Forms of Stress Invariants for Plasticity," J. of the Structural Division, ASCE, Vol. 98, pp. 949-954, 1972.

49. Nayak, G. C., "Plasticity and Large Deformation Problems by the Finite Element Method," umpublished Ph.D. Thesis, University of Wales, Swansea, 1971. (C/PH/15/1971).

50. Nayak, G. C. and Zienkiewicz, O.C., "Elasto-Plastic Stress Analysis, A Generalization for Various Constitutive Relations Including Strain Softening," International J. of Numerical Methods in Engineering, Vol. 5, pp. 113-135 (1972).

51. Nayak, G. C. and Zienkiewicz, O. C., "Note on the Alpha-Constant Stiffness Method for the Analysis of Nonlinear Problems," International J. of Numerical Methods in Engineering, Vol. 4, pp. 579-582 (1972).

52. Noor, A. K., "Nonlinear Analysis of Space Trusses," J. of the Structural Division, ASCE, Vol. 100, No. ST3, pp. 553-546, 1974.

53. Novozhilov, V. V., Foundations of the Nonlinear Theory of Elasticity, Graylock Press, 1953.

54. Oden, J. T., Finite Elements of Nonlinear Continua, McGraw-Hill, New York, 1972.

55. Pilkey, W., Saczalski, K. and Schaeffer, H. (Eds.) Structural Mechanics Computer Programs, University Press of Virginia, 1974.

56. Prager, W., and Hodge, P. G., Theory of Perfectly Plastic Solids, Wiley, New York, 1951.

57. Przemieniecki, J. S., Theory of Matrix Structural Analysis, McGraw-Hill, New York, 1968.

58. Rajasekaran, S. and Murray, D., "Incremental Finite Element Matrices," J. of the Structural Division, ASCE, Vol. 99, No. ST12, December 1973.

59. Rehak, D. R. and Lopez, L. A., "A Tool for Translating Problem Oriented Languages," to be published.

60. Reyes, S. F. and Deere, D. U., "Elasto-Plastic Analysis of Underground Openings by the Finite Element Method," Proceedings, First International Congress of Rock Mechanics, Vol. 11, pp. 477-486, Lisbon, 1966.

61. Roos, D., ICES--System Design, MIT Press, 1966.

62. Roos, D. and Miller, C. L., COGO-90 Engineering User's Manual," R64-12, Dept. of Civil Engineering, MIT, April 1964.

63. Rossow, E. C., Lo, D., and Chu, S., "Efficient Design-Analysis of Psysically Nonlinear Trusses," Proceedings, Sixth Electronic Computation Conference, August 7-9, 1974, Atlanta.

64. Salem, M. H. and Mohraz, B., "Nonlinear Analysis of Planar Reinforced Concrete Structures," Civil Engineering Studies, SRS No. 410, University of Illinois at Urbana-Champaign, July 1974.

65. Schrem, E., "Structural Aspects of Software Design for Nonlinear Finite Element Analysis," Formulations and Computational Algorithms in Finite Element Analysis: U.S.-German Symposium, MIT August 1976.

66. Schrem, E., "Development and Maintenance of Large Finite Element Systems," Structural Mechanics Computer Programs, Pilkey, Saczalski, and Schaeffer (Eds.), University Press of Virginia 1974.

67. Schrem, E., "From Program Systems to Programming Systems for Finite Element Analysis," Proceedings, U.S.-German Symposium of Formulations and Computational Methods in Finite Element Analysis, 1976, MIT Press.

68. Swanson, J. A., "ANSYS--Engineering Analysis System User's Manual," Swanson Analysis Systems, Inc., Elizabeth, Pa.

69. Tillerson, J. R., "A Treatise on Nonlinear Finite Element Analysis," unpublished Ph.D. thesis, A & M University, Texas 1973.

70. Tillerson, J. R., Stricklin, J. A., and Haisler, W. E., "Numerical Methods for the Solution of Nonlinear Problems in Structural Analysis," ASME, 1973 Annual Winter Meeting, Detroit, Michigan, November 11-15, 1973.

71. Wilson, E. L., "Solid SAP," UC SESM Report No. 71-19, University of California Berkeley, September 1970.

72. Zienkiewicz, O. C. and Nayak, G. C., " A General Approach to Problems of Plasticity and Large Deformation Using Isoparametric Elements," Proceedings, Conference on Matrix Methods in Structural Mechanics, Wright-Patterson AF Base, Ohio, 1971.

73. Zienkiewicz, O.C., The Finite Element Method in Engineering Science, McGraw-Hill, London, 1978.

74. Zienkiewicz, O. C., Valliappan, S., and King, I. P., "Elasto-Plastic Solutions of Engineering Problems -- 'Initial Stress,' Finite Element Approach," International J. Numerical Methods in Engineering, Vol. 1, pp. 75-100 (1969).

APPENDIX A

ELEMENT TANGENT STIFFNESSES

A.1 Formulation

In the Lagrangian formulation, forces exerted by a distorted element on its nodes are given by

$$\{IF\} = \int_V [B]^T \{\sigma\}\, dV \tag{A.1}$$

where the [B] matrix relates differential changes of Green strain to differential changes of element nodal displacements. $\{\sigma\}$ is the vector of 2nd Piola-Kirchoff stress components. Both [B] and $\{\sigma\}$ are functions of element nodal displacements and element local coordinates for general nonlinear analysis. Integration is performed over the initial configuration of the element. Components of {IF} are always directed along the undeformed local element axes.

The element tangent stiffness matrix, $[K_T]$, provides a linear relationship between differential internal forces and nodal displacements such that

$$d\{IF\} = [K_T]\, d\{U\} \tag{A.2}$$

where {U} is the vector of element nodal displacement components. The total differential of {IF} is given by

$$d\{IF\} = \int_V d[B]^T \{\sigma\}\, dV + \int_V [B]^T\, d\{\sigma\}\, dV \tag{A.3}$$

The second integral is readily expressed in terms of displacement differentials by the following well known substitutions

$$d\{\sigma\} = [D_T]\, d\{\varepsilon\} \tag{A.4}$$

$$d\{\varepsilon\} = [B]\, d\{U\} \tag{A.5}$$

which yield

$$\int_V [B]^T\, d\{\sigma\}\, dV = \left[\int_V [B]^T\, [D_T]\, [B]\, dV\right] d\{U\} \tag{A.6}$$

The right integral is termed the $[K^0]$ portion of the element tangent stiffness. For geometric nonlinear analysis, [B] in the above equation is a function of the nodal displacements {U}. For small displacement analysis, [B] is the familiar differential operator dependent on the element coordinates.

Computation of the element stiffness is conveniently split into nodal stiffness submatrices $[K_{ij}]$. Submatrix (ij) relates differential forces at node i to differential displacements at node j. The matrix $[K^0_{ij}]$ is then given by

$$[K^0_{ij}] = \int_V [B_i]^T\, [D_T]\, [B_j]\, dV \tag{A.7}$$

The three dimensional expression for $[B_i]$ including all nonlinear terms is given by (see Eq. 3.2.7)

$$[B_i] = \left[\begin{array}{c|c|c} X'_x\ N_x & Y'_x\ N_x & Z'_x\ N_x \\ \hline X'_y\ N_y & Y'_y\ N_y & Z'_y\ N_y \\ \hline X'_z\ N_z & Y'_z\ N_z & Z'_z\ N_z \\ \hline \begin{array}{c} X'_y\ N_x \\ + \\ X'_x\ N_y \end{array} & \begin{array}{c} Y'_y\ N_x \\ + \\ Y'_x\ N_y \end{array} & \begin{array}{c} Z'_y\ N_x \\ + \\ Z'_x\ N_y \end{array} \\ \hline \begin{array}{c} X'_x\ N_z \\ + \\ X'_z\ N_x \end{array} & \begin{array}{c} Y'_x\ N_z \\ + \\ Y'_z\ N_x \end{array} & \begin{array}{c} Z'_x\ N_z \\ + \\ Z'_z\ N_x \end{array} \\ \hline \begin{array}{c} X'_z\ N_y \\ + \\ X'_y\ N_z \end{array} & \begin{array}{c} Y'_z\ N_y \\ + \\ Y'_y\ N_z \end{array} & \begin{array}{c} Z'_z\ N_y \\ + \\ Z'_y\ N_z \end{array} \end{array}\right] \qquad \text{(A.8)}$$

where x, y, and z subscripts imply differentiation. All derivatives of N are for the i^{th} node shape function only.

Expansion of the first integral in Eq. A.3 requires an expression for $d[B_i]$. From the definition of deformed coordinates $\{X'\}$

$$\begin{aligned} X' &= \Sigma\ N_i\ (X_i + u_i) \\ Y' &= \Sigma\ N_i\ (Y_i + v_i) \qquad i = 1,2,\ldots\ nn \\ Z' &= \Sigma\ N_i\ (Z_i + w_i) \end{aligned} \qquad \text{(A.9)}$$

their derivatives are easily obtained. The number of element nodes is nn.

For example, X' is given by

$$X'_y = \frac{\partial X'}{\partial y} = \sum_1^{nn} \frac{\partial N_i}{\partial y} (X_i + u_i) \tag{A.10}$$

Dependence of $[B_i]$ on the nodal displacements becomes evident with substitution of all derivatives similar to Eq. A.10 into Eq. A.8. The $[B_i]$ matrix is split into two matrices, $[B_i^L]$ independent of nodal displacements, and $[B_i^{NL}]$, dependent on nodal displacements, by noting that cross derivatives of nodal coordinates are zero. For example,

$$\sum_1^{nn} \frac{\partial N_i}{\partial y} X_i = 0 \tag{A.11}$$

Similarly, $X_x = Y_y = Z_z = 1$. Thus, in taking differentials of $[B_i]$, $d[B_i^L] = 0$.

Differentials of each term in $[B_i^{NL}]$ are obtained using the chain rule. For example, $dB_{1,1}^{NL}$ is given by

$$dB_{1,1}^{NL} = \Sigma \frac{\partial B_{1,1}}{\partial u_j} du_j = \Sigma \frac{\partial}{\partial u_j} \left(\frac{\partial N_i}{\partial x} \frac{\partial N_i}{\partial x}\right) du_j \tag{A.12}$$

Substitution of $\Sigma(\partial N_k/\partial x)\, u_k = \partial u/\partial x$ and differentiating with respect to the nodal displacements yields

$$dB_{1,1}^{NL} = \frac{\partial N_i}{\partial x} \Sigma \frac{\partial}{\partial u_j} \left(\Sigma \frac{\partial N_k}{\partial x} u_k\right) du_j =$$

$$\frac{\partial N_i}{\partial x} \Sigma \frac{\partial N_j}{\partial x} du_j \tag{A.13}$$

If $\Sigma(\partial N_j/\partial x)du_j$ is denoted du_x, then $dB_{1,1}^{NL} = (\partial N_i/\partial x)du_x$.

Differentials of other $[B_i^{NL}]$ components are obtained in a similar way.

Differential internal force components at the i^{th} node can now be written more explicitly. For example, $d(IF_x^i)$ is given by

$$\begin{aligned} d(IF_x^i) = \int_V [&\sigma_x N_x\, du_x + \sigma_y N_y\, du_y + \sigma_z N_z\, du_z + \\ &\sigma_{xy} (N_x\, du_y + N_y\, du_x) + \\ &\sigma_{xz} (N_z\, du_x + N_x\, du_z) + \\ &\sigma_{yz} (N_y\, du_z + N_z\, du_y)\,]\, dV \end{aligned} \tag{A.14}$$

The differential force expressions must be factored into a matrix product to express in stiffness form. The terms du_x, du_y, etc. are first written in matrix form as

$$d\{\bar{U}\}_{9x1} = \Sigma\, [G_i]\, d\{U\}_j \tag{A.15}$$

where $d\{\bar{U}\}^T = [du_x, dv_x, dw_x, du_y, dv_y, dw_y, du_z, dv_z, dw_z]$ and

$$[G_i] = \begin{vmatrix} I_3 \frac{\partial N_i}{\partial x} \\ \hline I_3 \frac{\partial N_i}{\partial y} \\ \hline I_3 \frac{\partial N_i}{\partial z} \end{vmatrix} \tag{A.16}$$

I_3 is a 3 x 3 identity matrix. Define a matrix [M] such that

$$[M] = \left|\begin{array}{c|c|c} I_3\sigma_x & I_3\sigma_{xy} & I_3\sigma_{xz} \\ \hline I_3\sigma_{xy} & I_3\sigma_y & I_3\sigma_{yz} \\ \hline I_3\sigma_{xz} & I_3\sigma_{yz} & I_3\sigma_z \end{array}\right| \tag{A.17}$$

then, simple multiplication shows that differential forces at node i due to differential displacements at node j for the first integral of Eq. A.3 are given by

$$d\{IF_i\} = \left[\int_V [G_i]^T [M] [G_j] \, dV \right] d\{U_j\} \tag{A.18}$$

Therefore, the total element tangent stiffness submatrix (ij) is given by

$$[K_{T_{ij}}] = \int_V [G_i]^T [M] [G_j] + [B_i]^T [D_T] [B_j] \, dV \tag{A.19}$$

A.2 Truss Element

The three dimensional truss element with two nodes provides a simple example to illustrate computation of the element tangent stiffness. Only one local coordinate, X, is required. Displacements along the element in terms of nodal displacements are

$$\begin{aligned} U(X) &= N_1 U_1 + N_2 U_2 \\ V(X) &= N_1 V_1 + N_2 V_2 \\ W(X) &= N_1 W_1 + N_2 W_2 \end{aligned} \tag{A.20}$$

where the interpolation functions N_1 are defined by

$$\begin{aligned} N_1 (X) &= (L-X)/L \\ N_2 (X) &= X/L \end{aligned} \tag{A.21}$$

The $[G_1]$ and $[G_2]$ are then

$$\begin{aligned} [G_1] &= -1/L \, [I_3] \\ [G_2] &= 1/L \, [I_3] \end{aligned} \tag{A.22}$$

since all derivatives with respect to Y and Z vanish. Similarly, σ_x is the only non-zero stress component; thus,

$$[M] = \sigma_x [I_3] \tag{A.23}$$

The matrices $[B_1]$ and $[B_2]$ are simple as ε_x is the only strain component.

$$[B_i] = \left(\frac{\partial N_i}{\partial x}\right) [X'_x, Y'_x, Z'_x] \tag{A.24}$$

Computation of the deformed coordinates yields

$$X'_x = \frac{\partial}{\partial x} [N_1(X_1+U_1) + N_2(X_2+U_2)] = 1+(U_2-U_1)/L \tag{A.25}$$

$$Y'_x = (V_2 - V_1)/L$$

$$Z'_x = (W_2 - W_1)/L$$

Therefore,

$$[B_1] = -1/L\ [(1+(U_2-U_1)/L),\ (V_2-V_1)/L,\ (W_2-W_1)/L] \tag{A.26}$$

$$[B_2] = -[B_1]$$

For material nonlinearity, $[D_T]$ is a 1 x 1 matrix containing the uniaxial stress-strain curve slope, E_T. For this element, the [G] and [B] matrices are independent of the element coordinates, therefore the products in Eq. A.19 can be taken from under the integral sign. The resulting 6 x 6 tangent stiffness matrix is given below:

$$[K_T] = (A * \sigma_x)/L \begin{bmatrix} I_3 & -I_3 \\ -I_3 & I_3 \end{bmatrix} + (A * E_T)/L\ [B_1, B_2]^T \begin{bmatrix} B_1 \\ B_2 \end{bmatrix} \tag{A.27}$$

A.3 Truss Element Implementation in FINITE

To complete the presentation of element tangent stiffness generation, the library input tables and FORTRAN subroutine for the three dimensional truss element are given. Fig. A.1 lists the POL statements required to define the spacetruss element to FINITE. Most of this data is self-documenting; detailed explanations are therefore omitted.

The FORTRAN subroutine called by FINITE to generate the tangent stiffness matrix is shown in Fig. A.2. This routine generates the tangent stiffness for a rod, planetruss, or spacetruss element. The labeled COMMON area in FINITE element routines minimizes the number of local variables. Variables in this COMMON space are not passed between element routines.

APPENDIX B

PLASTICITY MATERIAL MODELS

B.1 General

Two yield functions commonly used in plasticity material models were implemented in FINITE during this work. The Von Mises yield criterion with isotropic hardening adequately predicts the behavior of many ductile metals subjected to monotonic loading. The Drucker-Prager criterion [18] approximates the Mohr-Coulomb yield function and is relevant for some cohesive materials including concrete and rock. Both models are standard in most nonlinear finite element systems and are used in this work to a) verify the material modeling processors of FINITE, and b) demonstrate the system's problem solving capability. A brief review of the formulation and computational procedures is presented as well as listings of the tables and subprograms for incorporating the models into FINITE. Complete discussions of plasticity theory as oriented toward finite element applications can be found in Refs. [11, 48, 49, 50, 74].

The implementation of these models in FINITE is readily adaptable to other yield criterion and associative flow rules that follow general plasticity theory. Special numerical techniques used here include the "subincrement" method that prevents artificial hardening due to replacement of differential quantities by finite increments. Additional corrective techniques are employed during the transition between elastic and plastic states.

B.2 Formulation

Experimental results have verified the negligible effect of hydrostatic stress on yielding in ductile metals. Von Mises proposed that

yielding of a material under a complex state of stress is linked to the distortional strain energy density, i.e., energy due to change of shape not change of volume. At initiation of yielding, the distortional strain energy under complex stress states is assumed equal to that for simple shear. Equating the two expressions for strain energy defines a yield function given by

$$F_{VM} = \sqrt{3}\, J_2^{1/2} - \bar{\sigma} \tag{B.1}$$

where,

$$J_2 = \frac{1}{2}(S_x^2 + S_y^2 + S_z^2) + \tau_{xy}^2 + \tau_{xz}^2 + \tau_{yz}^2 \tag{B.2}$$

and

$$\begin{aligned} S_x &= \sigma_x - \sigma_m \\ S_y &= \sigma_y - \sigma_m \\ S_z &= \sigma_z - \sigma_m \end{aligned} \tag{B.3}$$

$$\sigma_m = (\sigma_x + \sigma_y + \sigma_z)/3 \tag{B.4}$$

$\bar{\sigma}$ is the yield stress of the material in uniaxial tension. Expressed in terms of principal stresses, F_{VM} defines a cylindrical surface in stress space with the generator along the line $\sigma_1 = \sigma_2 = \sigma_3$ and radius of length $\bar{\sigma}\sqrt{2/3}$. The intersection of the cylinder with the $\sigma_1\sigma_2$ plane defines the well known Von Mises ellipse for two dimensional states of stress. These surfaces are shown in Fig. B.1.

The Drucker-Prager yield function is similar to the Von Mises criterion but incorporates the effect of hydrostatic stress. The yield function is

expressed by

$$F_{DP} = 3\alpha\ \sigma_m + J_2 - K \tag{B.5}$$

where α and K are material constants dependent upon the cohesion strength, C, obtained from an unconfined compression test and Φ, the angle of internal friction. The Drucker-Prager yield function plots as a right circular cone with the same generator axis as the Von Mises cylinder and approximates the Mohr-Coulomb pyramid yield surface. The cone diameter is proportional to the hydrostatic compression.

Three sets of coefficients α and K have been proposed [60]. The first set is obtained by forcing the cone to pass through edges of the Mohr-Coulomb pyramid determined from a triaxial test. The second coefficients approximate the pyramid for the axial extension test. The coefficients for these two sets are

$$\alpha_1 = \frac{2\ \text{SIN}\ \Phi}{\sqrt{3}\ (3\text{-SIN}\ \Phi)} \qquad K_1 = \frac{6\ C\ \text{COS}\ \Phi}{\sqrt{3}\ (3\text{-SIN}\ \Phi)} \tag{B.6}$$

$$\alpha_2 = \frac{2\ \text{SIN}\ \Phi}{\sqrt{3}\ (3\text{+SIN}\ \Phi)} \qquad K_2 = \frac{6\ C\ \text{COS}\ \Phi}{\sqrt{3}\ (3\text{+SIN}\ \Phi)} \tag{B.7}$$

Drucker and Prager gave the following constants for the special case of plane strain

$$\alpha_3 = \frac{\text{TAN}\ \Phi}{(9 + 12\text{TAN}^2\ \Phi)^{1/2}} \qquad K_3 = \frac{3\ C}{(9 + 12\text{TAN}^2\ \Phi)^{1/2}} \tag{B.8}$$

For given values of C and Φ, the constants of B.8 have the least values of α and K and therefore are the more conservative. The analyst may choose any set of coefficients through the material model property COEFFICIENTS. Values 1, 2, and 3 for this property correspond to coefficients defined by Eqs. B.6, B.7, and B.8 respectively.

All possible elastic stress states lie inside the cylinder and cone ($F < 0$). Stress states on the yield surface ($F = 0$) correspond to yielding or a plastic condition. Materials cannot maintain stress states outside the yield surface ($F > 0$).

Once the stress state reaches the yield surface during incremental loading, a flow rule is required to relate subsequent changes in stress to increments of strain. Most theories assume a linear relationship between infinitesimal stress and strain increments. By examining the implications of work hardening and the ideally plastic material, Drucker concluded that plastic strain increments are proportional to the gradient of the yield surface. Thus,

$$\{d\varepsilon_p\} = d\lambda \{a\} \tag{B.9}$$

where $\{a\}$ contains components of the outward normal to the yield surface in stress space and $d\lambda$ is a positive constant. This relation is termed the normality principle.

A hardening rule is required to predict changes in the yield surface due to plastic deformation of the material. As work hardening enables the material to maintain higher states of stress, the yield surface must be modified accordingly to keep the stress state on the surface. The simplest form of hardening assumes that the yield surface retains its original shape and expands uniformly to reflect the new yield stress. This type hardening rule is termed isotropic and ignores the Bauschinger effect. Other hardening rules distort the yield cylinder into a cone, for example, or rotate and translate the cylinder as a rigid body in stress space. For the Von Mises model, isotropic hardening is achieved by

equating the plastic work done by actual stresses to the plastic work done by a specimen of the material under uniaxial tension. No strain hardening is permitted for the Drucker-Prager model implemented here.

With the yield function, flow and hardening rules established, derivation of the constitutive relation valid in the plastic range is relatively straightforward. For a given infinitesimal increment of strain, $\{d\varepsilon\}$, composed of elastic, $\{d\varepsilon_e\}$, and plastic, $\{d\varepsilon_p\}$, portions the following relations hold

$$\{d\sigma\} = [D]\ \{d\varepsilon_e\} \tag{B.10}$$

$$\{d\sigma\} = [D]\ \{d\varepsilon\} - [D]\ \{d\varepsilon_p\} \tag{B.11}$$

where $[D]$ is usual matrix of elastic constants for isotropic materials and $\{d\varepsilon_p\}$ is computed by Eq. B.9. When stresses are such that $F = 0$, the differential must also be zero. In the Von Mises criterion, for example,

$$F = dF = \{\frac{\partial F}{\partial \sigma}\}\ \{d\sigma\}\ - d\bar{\sigma} = 0 \tag{B.12}$$

or

$$d\bar{\sigma} = \{\frac{\partial F}{\partial \sigma}\}\ \{d\sigma\} \tag{B.13}$$

where $\{\partial F/\partial \sigma\}$ is the stress gradient normal to the yield surface. Thus, $\{a\}$ of Eq. B.9 is given by $\{\partial F/\partial \sigma\}$. Equating the plastic work done by the stress state causing yield to the plastic work done by the uniaxial yield stress shows that

$$\{\bar{\sigma}\}\ \{d\varepsilon_p\} = \bar{\sigma}\ d\bar{\varepsilon}_p \tag{B.14}$$

Substitution of the plastic strain expression from the flow rule, Eq. B.9, yields

$$d\lambda \{\sigma\}^T \{a\} = \bar{\sigma}\, d\varepsilon_p \tag{B.15}$$

Euler's theorem for homogeneous functions shows that $\{\sigma\}^T\{a\} = 0$ from which

$$d\lambda = d\bar{\varepsilon}_p \tag{B.16}$$

Pre-multiplying Eq. B.11 by $\{a\}^T$ gives

$$\{a\}^T \{d\sigma\} = d\bar{\sigma} =$$

$$\{a\}^T [D] \{d\varepsilon\} - \{a\}^T [D]\, d\bar{\varepsilon}_p \{a\} \tag{B.17}$$

The increment of uniaxial stress, $d\bar{\sigma}$, is given by

$$d\bar{\sigma} = H'\, d\bar{\varepsilon}_p \tag{B.18}$$

where H' is the slope of the uniaxial tension-plastic strain curve obtained from test results. Substituting into Eq. B.17 and solving for $d\varepsilon_p$,

$$d\lambda = d\bar{\varepsilon}_p = \frac{\{a\}^T [D]}{H' + \{a\}^T [D] \{a\}} \{d\varepsilon\} \tag{B.19}$$

Substitution of this expression into Eq. B.11 provides the desired relation between stress and strain in the plastic range

$$\{d\sigma\} = [D_T] \{d\varepsilon\}$$

$$[D_T] = [D] - \frac{[D] \{a\}\{a\}^T [D]}{H' + \{a\}^T [D] \{a\}} \tag{B.20}$$

The above expressions are used to evaluate the change in stress state for a finite increment of strain $\{\Delta\varepsilon\}$ as follows

$$\{\Delta\sigma\} = R\,[D]\,\{\Delta\varepsilon\} + \int_{R\Delta\varepsilon}^{\Delta\varepsilon} [D_T]\,\{d\varepsilon\} \qquad (B.21)$$

where R is the portion of the strain increment required to reach the yield surface from a previously elastic stress state. If the previous stress state was on the yield surface, R = 0. For small increments of load resulting in small strain increments, the above expression is approximately equal to

$$\{\Delta\sigma\} = R\,[D]\,\{\Delta\varepsilon\} + (1-R)\,[D_T]\,\{\Delta\varepsilon\} \qquad (B.22)$$

This approximation is inadequate for large strain increments and causes drifting from the yield surface thereby introducing artificial hardening as shown in Fig. B.3. Special numerical techniques to reduce drifting effects are described in the next section.

B.3 Numerical Refinements

The R Factor for transitioning between elastic and plastic stress states is computed more accurately by the process outlined below applicable for all convex yield surfaces. This procedure was originally developed by Nayak [50]. Let $\{\sigma_0\}$ denote an initial stress state inside the yield surface as shown in Fig. B.3. If the entire strain increment is considered elastic, the new stress state is outside the yield surface. This condition is expressed by

$$\begin{aligned} F(\{\sigma_0\}) &= F_0 < 0 \\ \{\Delta\sigma_e\} &= [D]\,\{\Delta\varepsilon\} \qquad (B.23) \\ F(\{\sigma_0\} + \{\Delta\sigma_e\}) &= F_1 > 0 \end{aligned}$$

The R factor is chosen such that

$$F(\{\sigma_0\} + R\{\Delta\sigma_e\}) = 0 \tag{B.24}$$

Simple linear interpolation between F_0 and F_1 provides the first estimate for R

$$R_1 = -F_1/(F_1 - F_0) \tag{B.25}$$

But because F is a nonlinear function of the stress, the correction is not exact,

$$F(\{\sigma_0\} + R_1\{\Delta\sigma_e\}) = F_2 \neq 0 \tag{B.26}$$

A small change in R_1 is necessary to produce a change in F, i.e., F + dF = 0. For an instantaneous position of the yield surface, dF is given by $\{a\}^T\{d\sigma\}$ with a evaluated at the stress state $\{\sigma_0\} + R_1\{\Delta\sigma_e\}$. Approximating $\{d\sigma\}$ by $\Delta R_1\{\Delta\sigma_e\}$, the change in R_1 is given by

$$\Delta R_1 = -F_2/(\{a\}^T\{\Delta\sigma_e\}) \tag{B.27}$$

with the improved value for R given by $R = R_1 + \Delta R_1$. This process has a simple geometric interpretation as shown in Fig. B.3. The projection of the estimated stress state to cause yield, computed using R_1, onto the yield surface normal is forced to zero by the correction dF.

Drift from the yield surface caused by approximating the integral of Eq. B.21 by the expression in Eq. B.22 is minimized with the "subincrement" procedure. The technique is based on the geometrical interpretation of the $[D_T]$ matrix as shown in Fig. B.3. This matrix forces the new stress state to remain on the yield surface for infinitesimal strain increments.

When finite, rather than infinitesimal, strain increments, occur during actual problem solution considerable drift from the yield surface may result. To minimize drift, the portion of the strain increment not taken elastically, (1-R) $\{\Delta\varepsilon\}$, is subdivided into M equal increments. The integral in Eq. B.21 is then approximated by

$$\int_{R\Delta\varepsilon}^{\Delta\varepsilon} [D_T]\ \{d\varepsilon\} = (1-R) \sum_{1}^{M} [D_T]\{\Delta\varepsilon\}/M \tag{B.28}$$

in which $[D_T]$ is updated at the end of each subincrement. The number of subincrements is based upon the magnitude of the initial deviation from the yield surface for the entire strain increment taken elastically or the deviation computed using the $[D_T]$ incorporated in the current element stiffness matrix if the material has previously yielded. If the initial deviation is denoted F_1, then the number of increments is given by

$$\begin{aligned} M &= F_1/(\alpha\bar{\sigma}) < M_{max} \text{ (Von Mises)} \\ M &= F_1/(\alpha K) < M_{max} \text{ (Drucker-Prager)} \end{aligned} \tag{B.29}$$

where α is a fraction of the current yield stress. For small deviations only a few subincrements are required. A limit on the maximum number of increments is necessary to detect instability of the procedure (a large number of subincrements indicates overall analysis divergence).

Three user supplied material model properties control the subincrement procedure. The property ALPHA corresponds to α in Eq. B.29 and has a default value of 0.05. The maximum number of increments is given by the property MAXINCR and has a default value of 30. DIVERGE is the number of

increments above which the solution is said to be diverging (default value of 500). If the number of subincrements is greater than MAXINCR but less than DIVERGE, the number of increments used is MAXINCR. One final property, TOLERANCE, is used to determine if a state of stress is on the yield surface F. If the yield function F has an absolute value less than TOLERANCE * $\bar{\sigma}$ for the Von Mises criterion or TOLERANCE * K for the Drucker-Prager criterion, then the state of stress is assumed to lie on the yield surface. The default value of TOLERANCE is 0.001.

Even with the subincrement process, updated stresses at the end of each subincrement will not satisfy the yield criterion. Let $\{\sigma'\}$ denote updated stresses following each subincrement, then

$$F(\{\sigma'\}) = F_1 \neq 0 \tag{B.30}$$

A final corrective stress $\{\Delta\sigma'\}$ computed by a procedure similar to that in Eq. B.27 is

$$\{\Delta\sigma'\} = -\{a\}\ (F_1/\{a\}^T\{a\}) \tag{B.31}$$

where $\{a\}$ is evaluated for the stresses $\{\sigma'\}$.

B.4 Computational Procedure

The major steps for updating stresses given an increment of strain are summarized below.

1) Subtract that portion of the incremental strain vector due to initial strain effects to obtain the increment of mechanical strain, $\{\Delta\varepsilon\}$. Using the constitutive matrix for the point incorporated in the current element stiffness, either [D] or

$[D_T]$, compute the trial stress increment $\{\Delta\sigma_e\}$. Add these to the previous total stresses to obtain a new trial stress $\{\sigma'\} = \{\sigma_0\} + \{\Delta\sigma_e\}$.

2) Evaluate the yield function $F(\{\sigma'\}) = F_1$. If the previous state of stress was plastic, $F(\{\sigma_0\}) = F_0 = 0$, set R = 0 and go to step 4. If $F_0 < 0$, the new stress state is still elastic. Trial stresses $\{\sigma'\}$ are actual stresses. Omit remaining steps. If $F_1 > 0$ and $F_0 < 0$, the R factor is required. Go to the next step.

3) Compute $R_1 = -F_1/(F_1-F_0)$. Add $R_1\{\Delta\sigma_e\}$ to $\{\sigma_0\}$ and re-evaluate the yield function to obtain F_2. Determine $\{a\}$ and compute $R = R_1-F_2/\{a\}^T\{\Delta\sigma_e\}$. Generate the new trial stress as $\{\sigma'\} = \{\sigma_0\} + R\{\Delta\sigma_e\}$. The yield function should be very near zero for these stresses.

4) Compute M, the number of subincrements. $M = F_1/(\alpha\sigma) + 1$ or $M = F_1/(\alpha K) + 1$. Compute the subincrement strain increment, $\{\Delta\varepsilon"\} = \{\Delta\varepsilon\}/M$, and the corresponding elastic stress increment, $\{\Delta\sigma"\} = [D]\{\Delta\varepsilon"\}$. Repeat steps 5-7, M times.

5) Invoke the material stress-strain function to return H' for the total accumulated uniaxial plastic strain. Omit this step for the Drucker-Prager model.

6) For the current stresses, $\{\sigma'\}$, compute $\{a\}$ and the plastic multiplier $d\lambda$ using Eq. B.19. If $d\lambda < 0$, the material is unloading elastically from a plastic state. Compute the new stress using the [D] matrix and omit remaining subincrements.

For positive $d\lambda$, compute the plastic strain increments, $\{\Delta\varepsilon_p\}$, and adjust the elastic increment $\{\Delta\sigma''\}$ to reflect them. $\{\sigma'\} = \{\sigma'\} + \{\Delta\sigma''\} - d\lambda[D]\{a\}$.

7) Update the uniaxial plastic strain to include $d\lambda$. If $H' = 0$, adjust stresses using Eq. B.31. For a strain hardening material, compute the new yield stress $\bar{\sigma}$ by evaluating $F(\{\sigma'\})$ with previous $\bar{\sigma} = 0$.

8) The final new stress state for the material is that computed for the last subincrement. Update the material history parameters to indicate the state is plastic and save the accumulated uniaxial plastic strain.

B.5 Implementation in FINITE

The POL statements to define the Von Mises material model and the SEGMENTAL stress-strain function are shown in Figs. B.4 and B.5. The two corresponding FORTRAN subroutines are listed in Figs. B.6 and B.7.

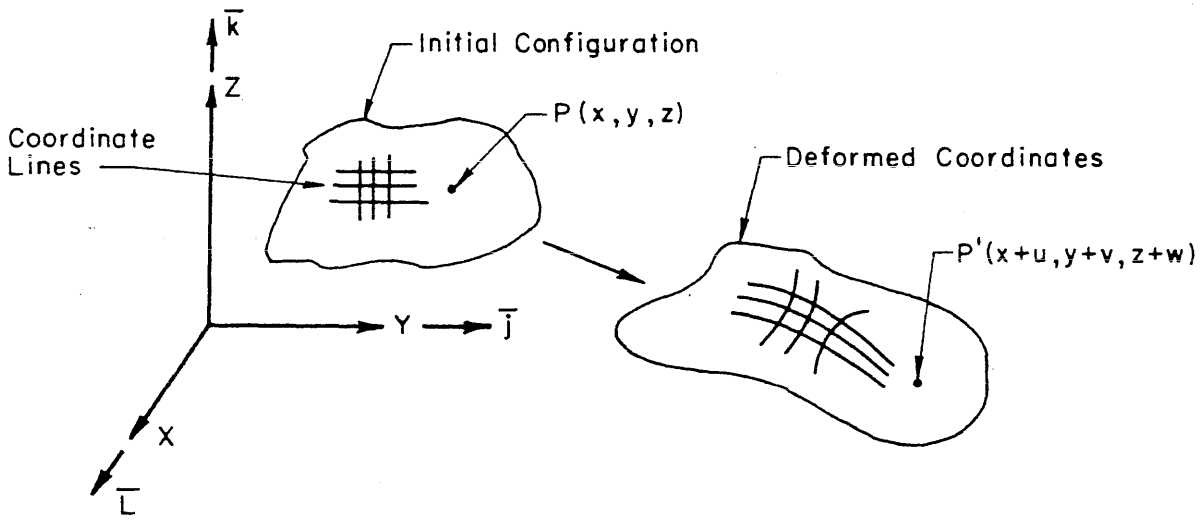
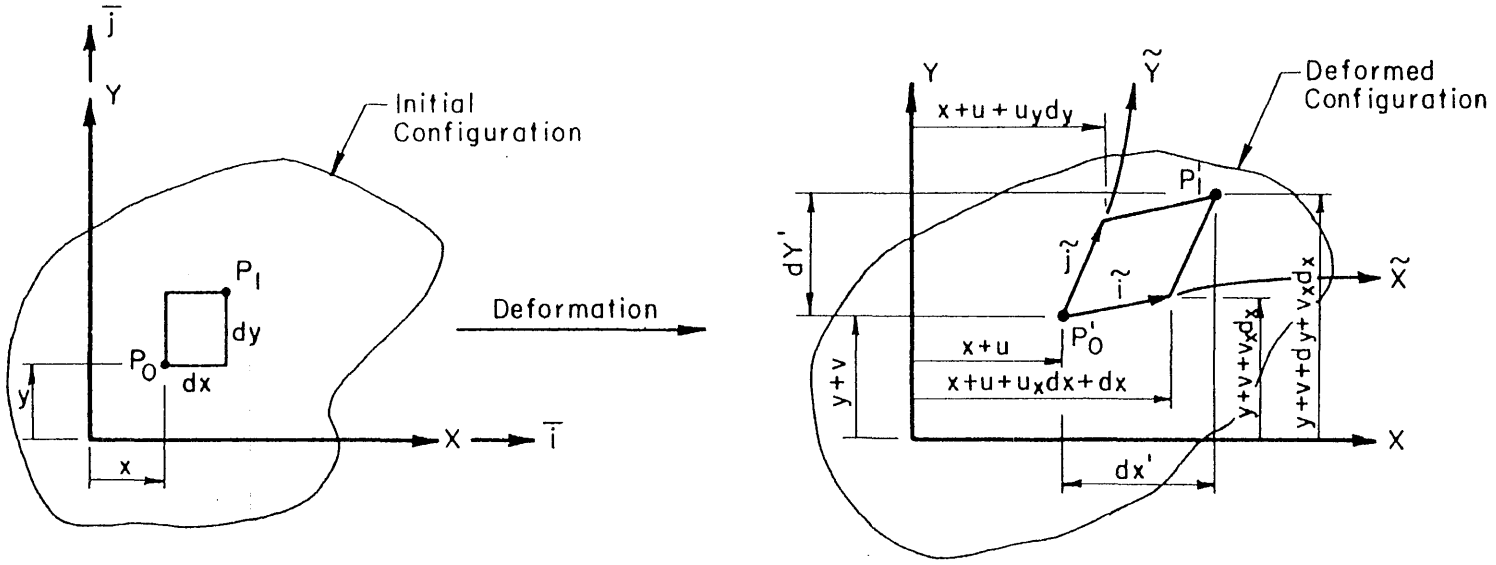


Fig. 2.2.1. Initial and Deformed Coordinates



$$\tilde{i} = (1+u_x)\,dx\,\bar{i} + v_x dx\,\bar{j}$$
$$\tilde{j} = u_y dy\,\bar{i} + (1+v_y)\,dy\,\bar{j}$$

Fig. 2.2.2. Two Dimensional Deformation

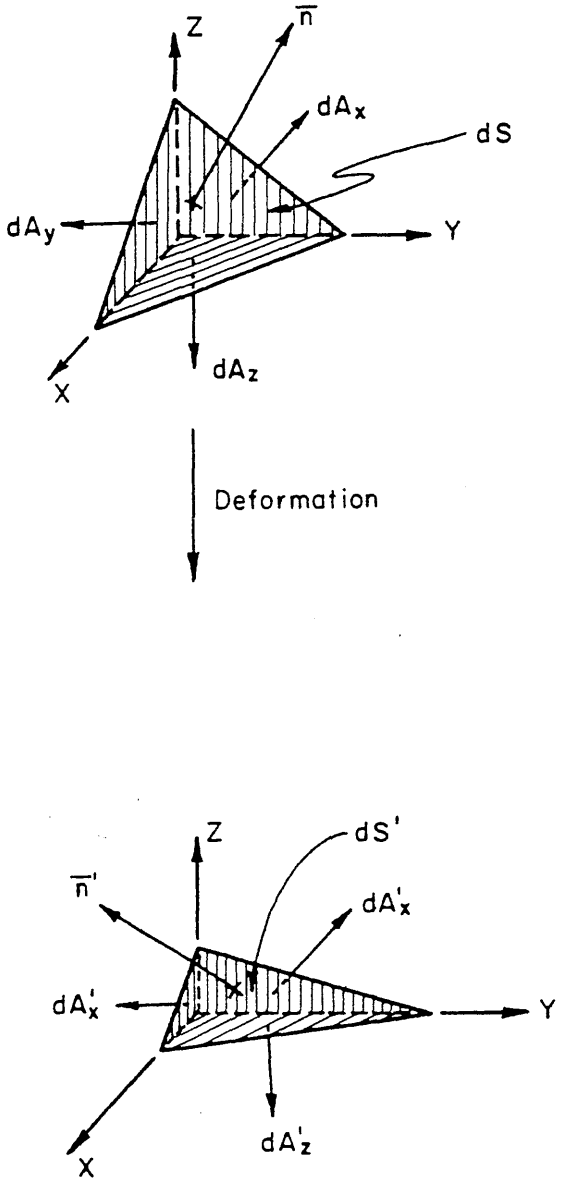


Fig. 2.2.3. Initial and Deformed Area Projections

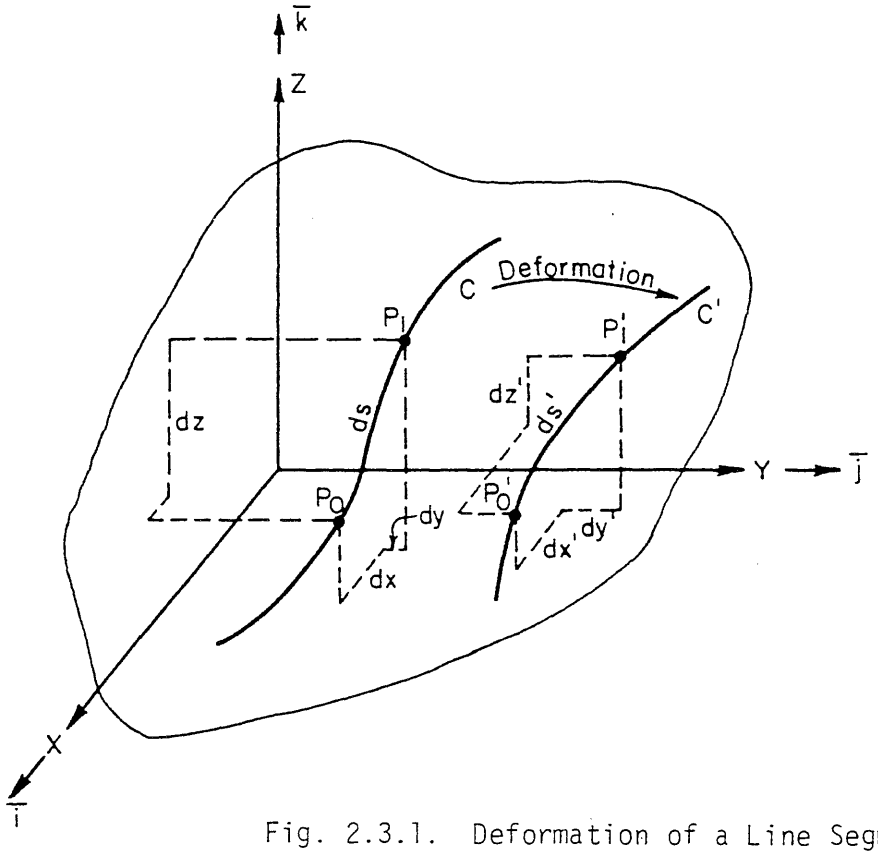
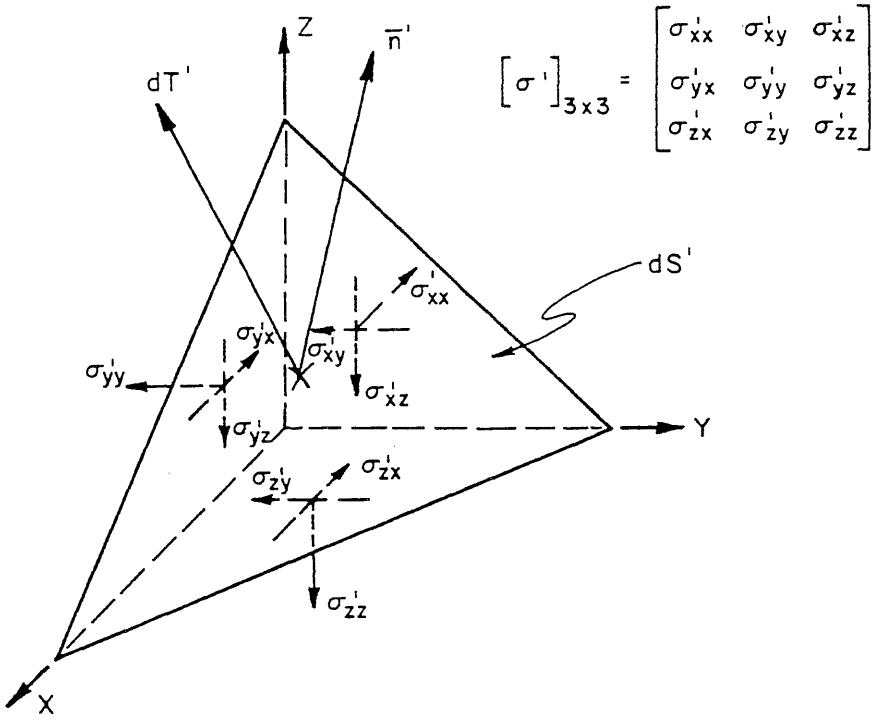


Fig. 2.3.1. Deformation of a Line Segment



$$\left[\sigma'\right]_{3\times3} = \begin{bmatrix} \sigma'_{xx} & \sigma'_{xy} & \sigma'_{xz} \\ \sigma'_{yx} & \sigma'_{yy} & \sigma'_{yz} \\ \sigma'_{zx} & \sigma'_{zy} & \sigma'_{zz} \end{bmatrix}$$

Fig. 2.4.1. Eulerian Stress Components

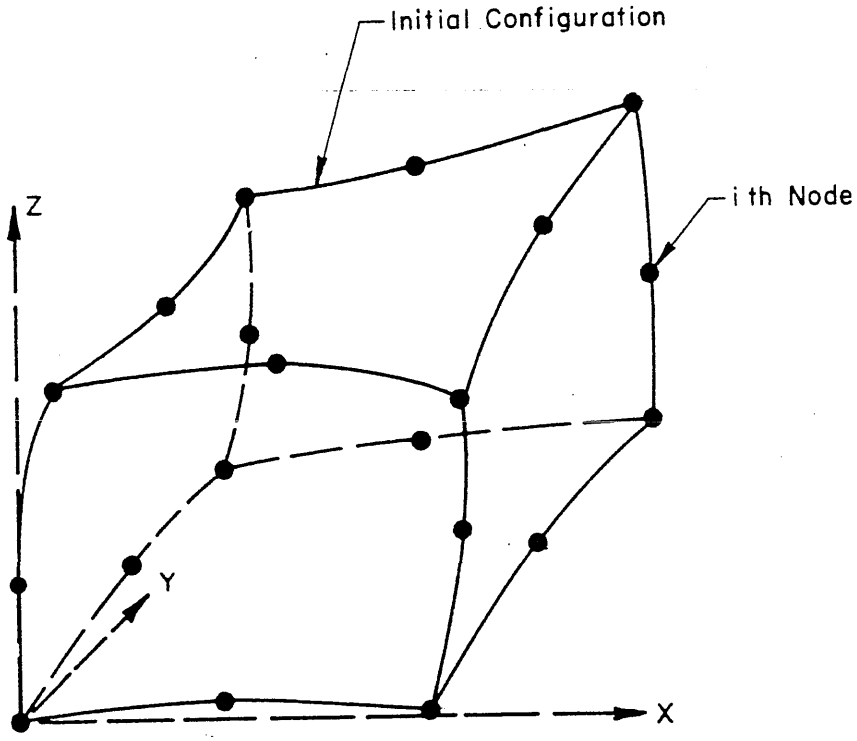


Fig. 3.2.1. Element Local Coordinates

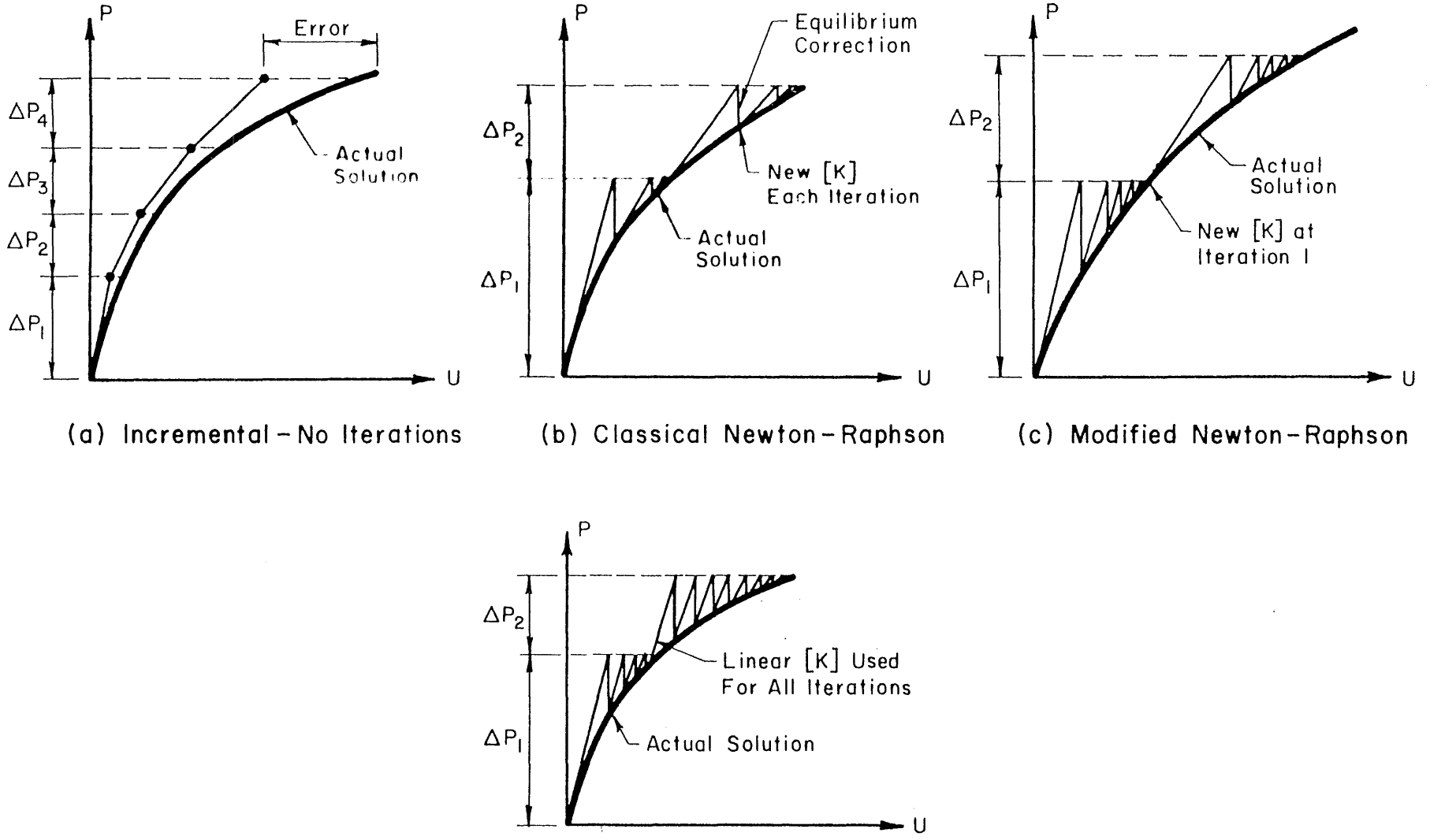


Fig. 3.4.1. Newton-Raphson Solution Procedures

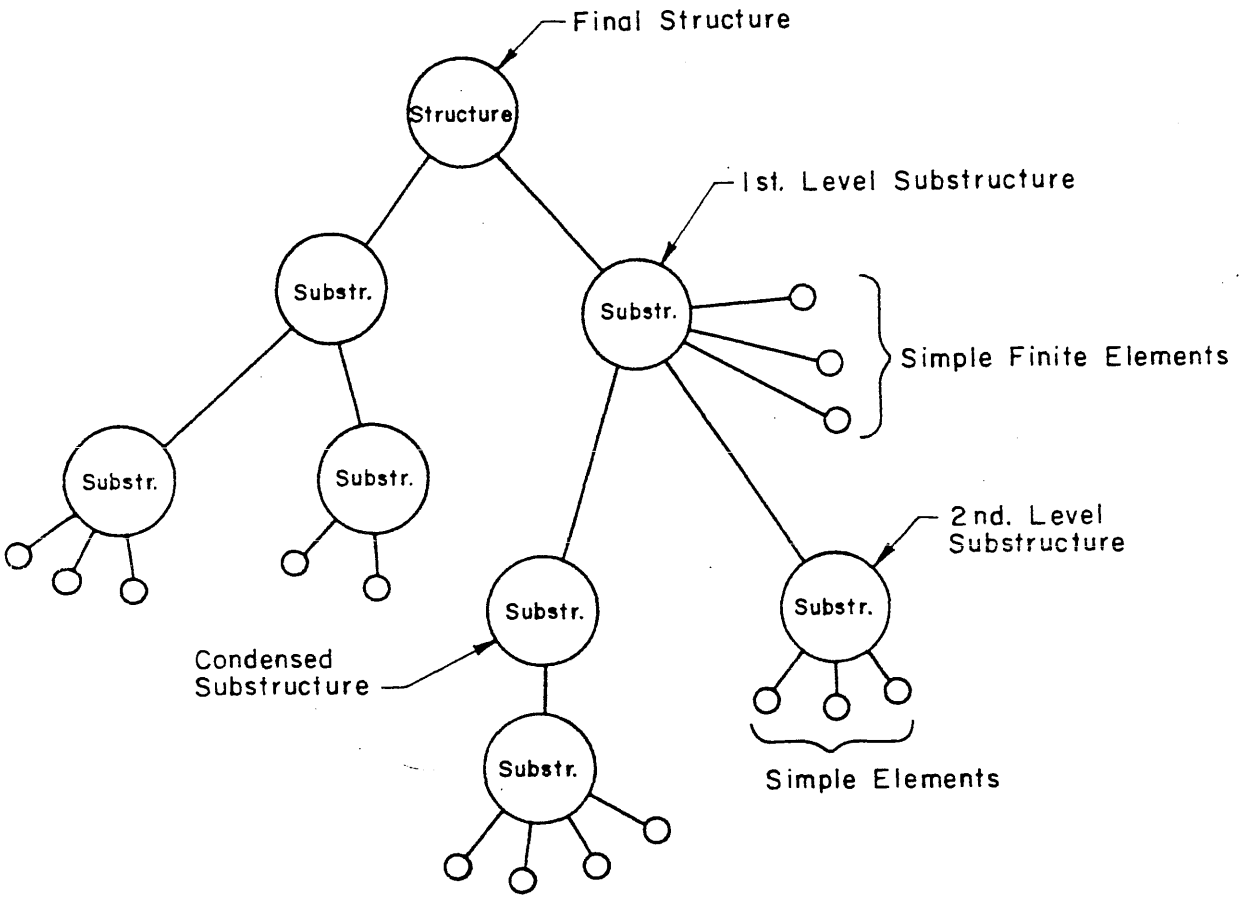


Fig. 3.6.1. Structural Hierarchy

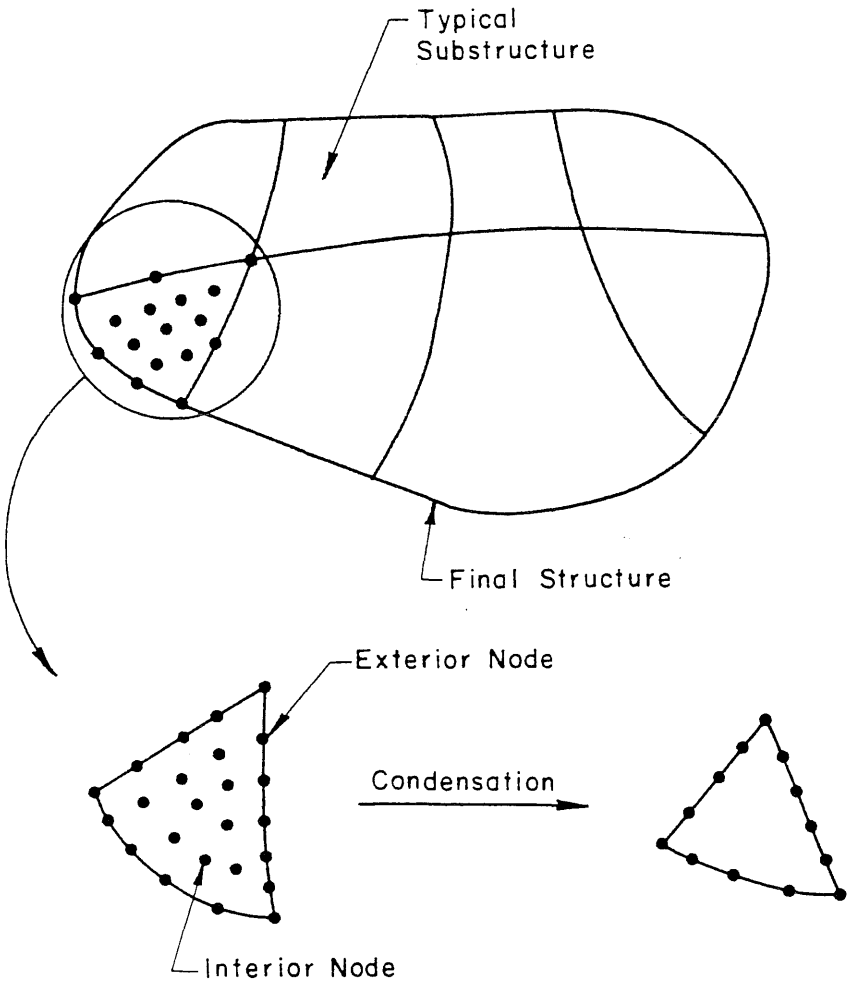


Fig. 3.6.2. Static Condensation of Internal Substructure Nodes

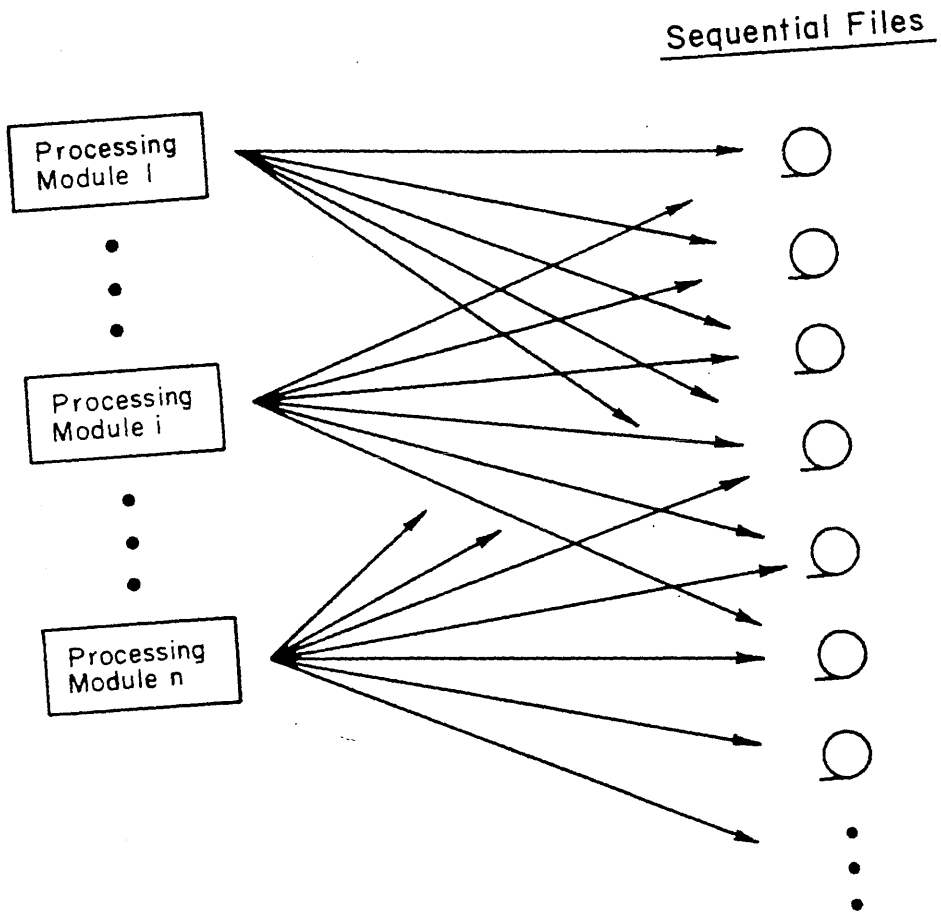


Fig. 5.2.1. Typical Nonlinear Software Organization

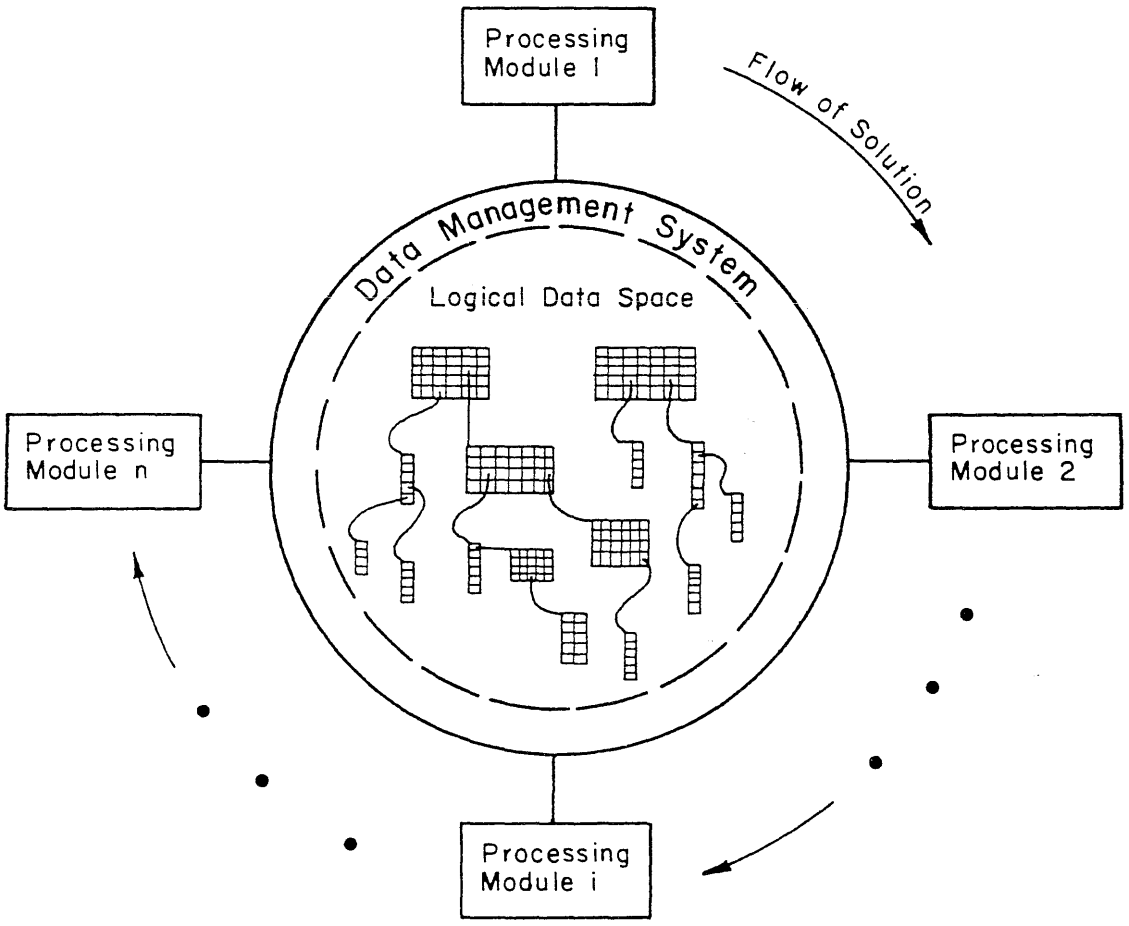


Fig. 5.2.2. Nonlinear Finite Element Software Organized Around a Data Base Manager

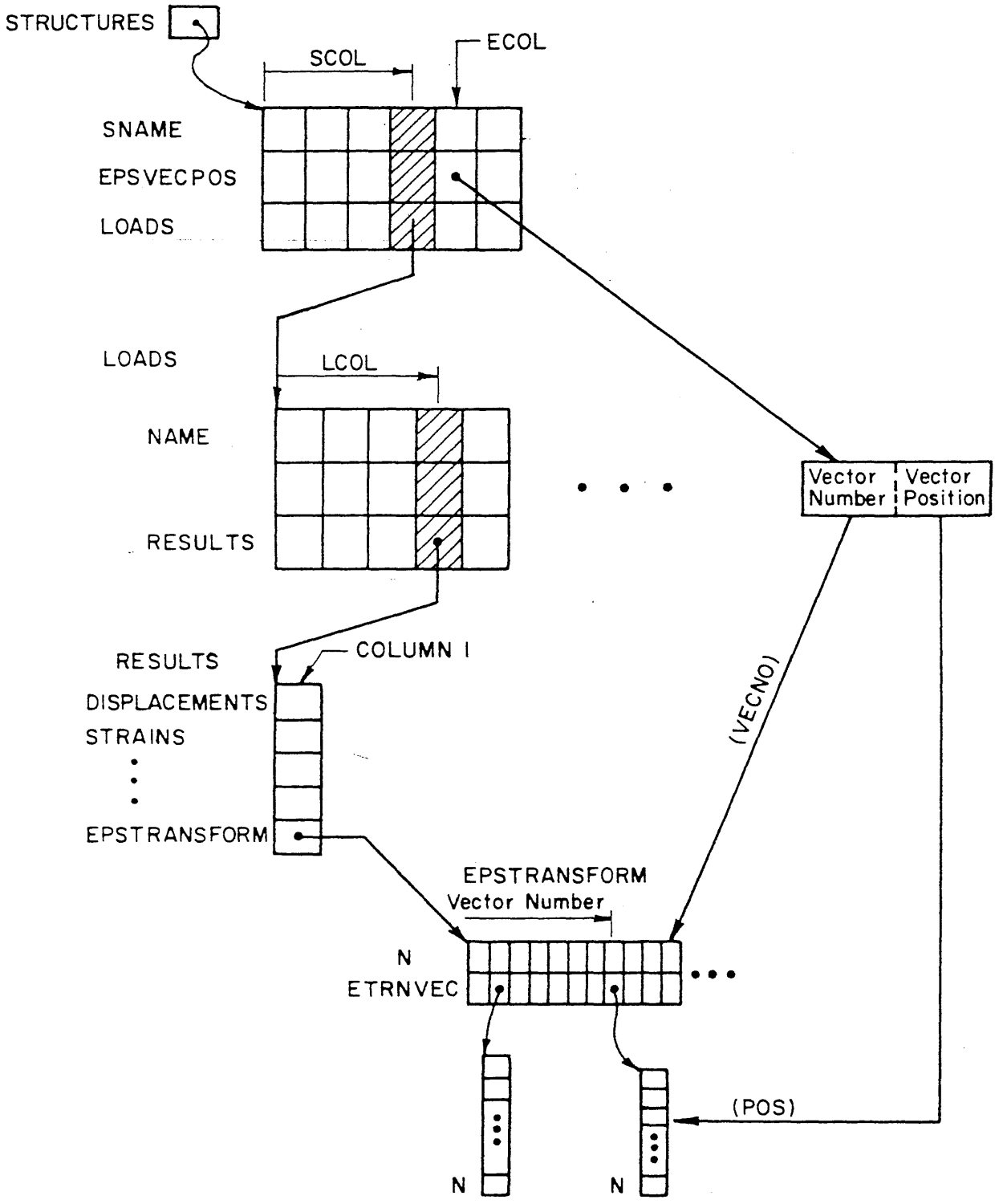


Fig. 5.3.1. Typical Hierarchial Data Structure

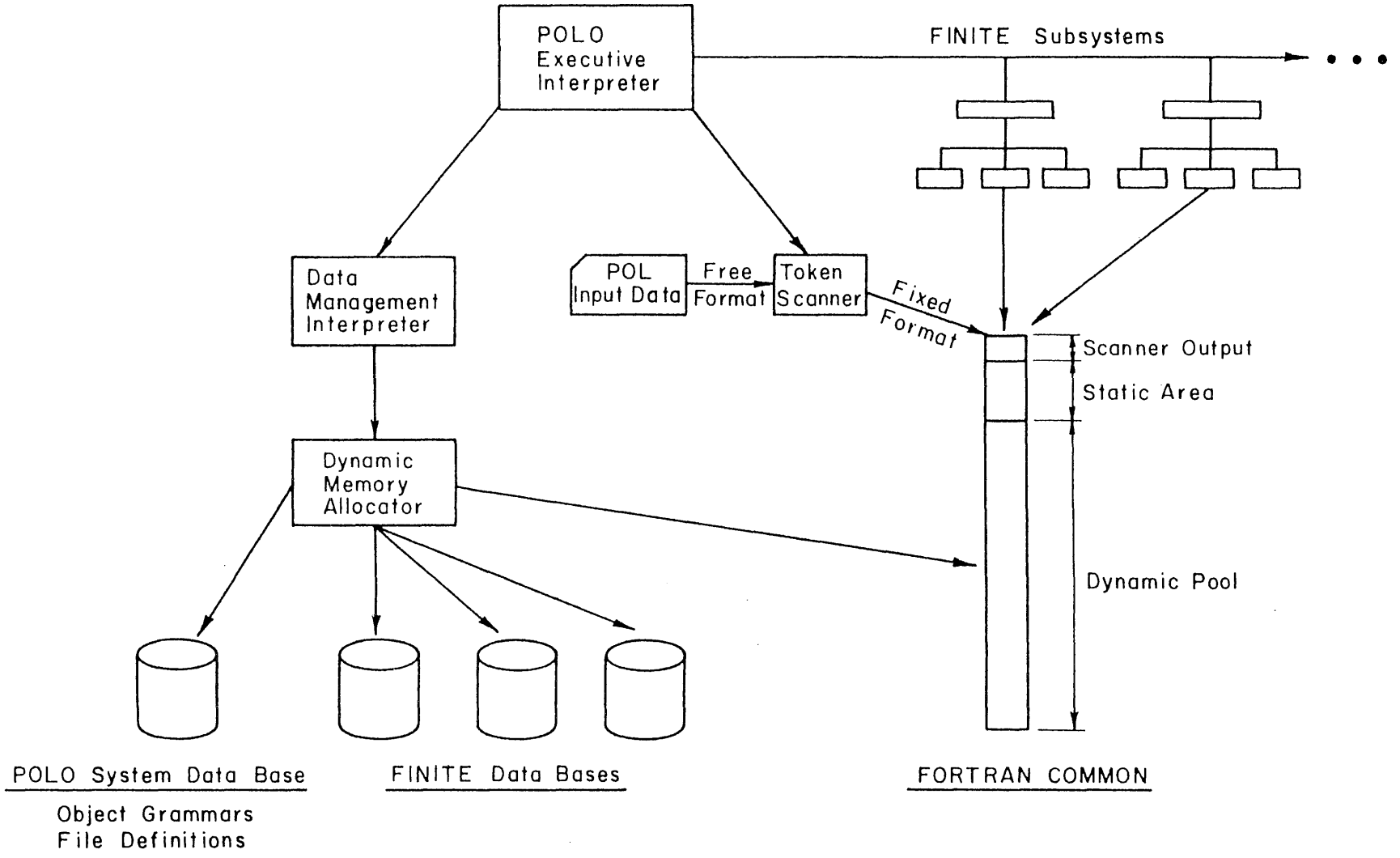


Fig. 5.3.2. POLO-FINITE Configuration During Execution

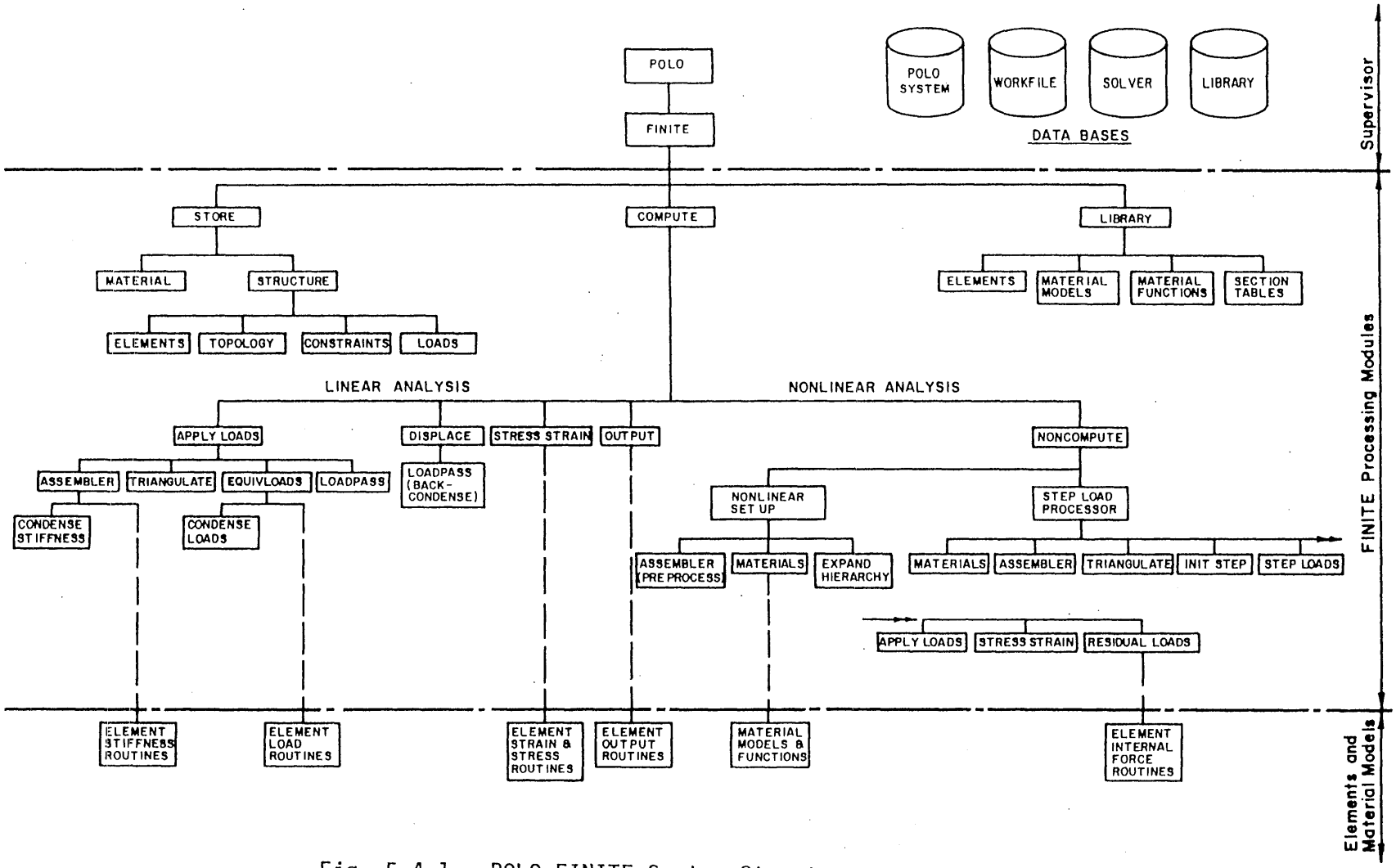


Fig. 5.4.1. POLO-FINITE System Structure

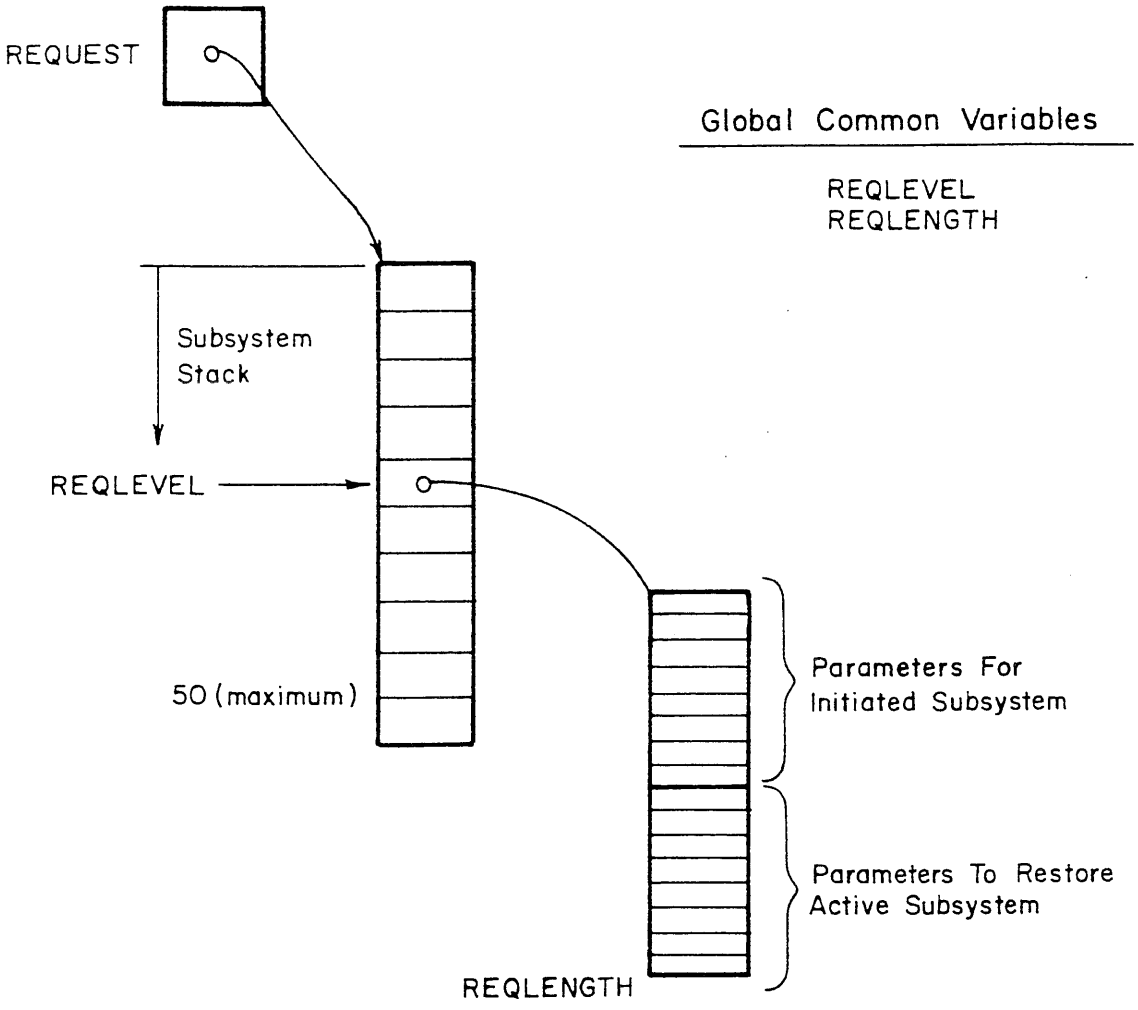


Fig. 5.4.2. Data Structure for Interfacing FINITE Subsystems

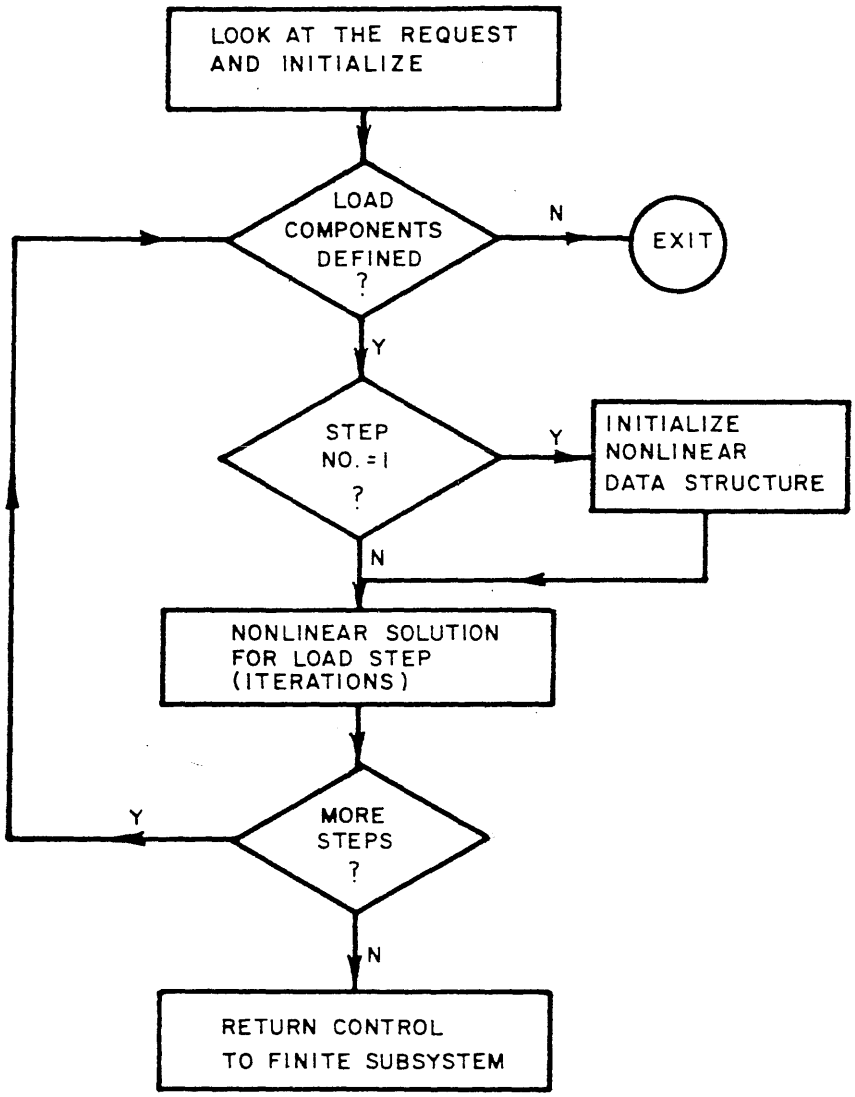


Fig. 5.6.1. Overview of Solution Process

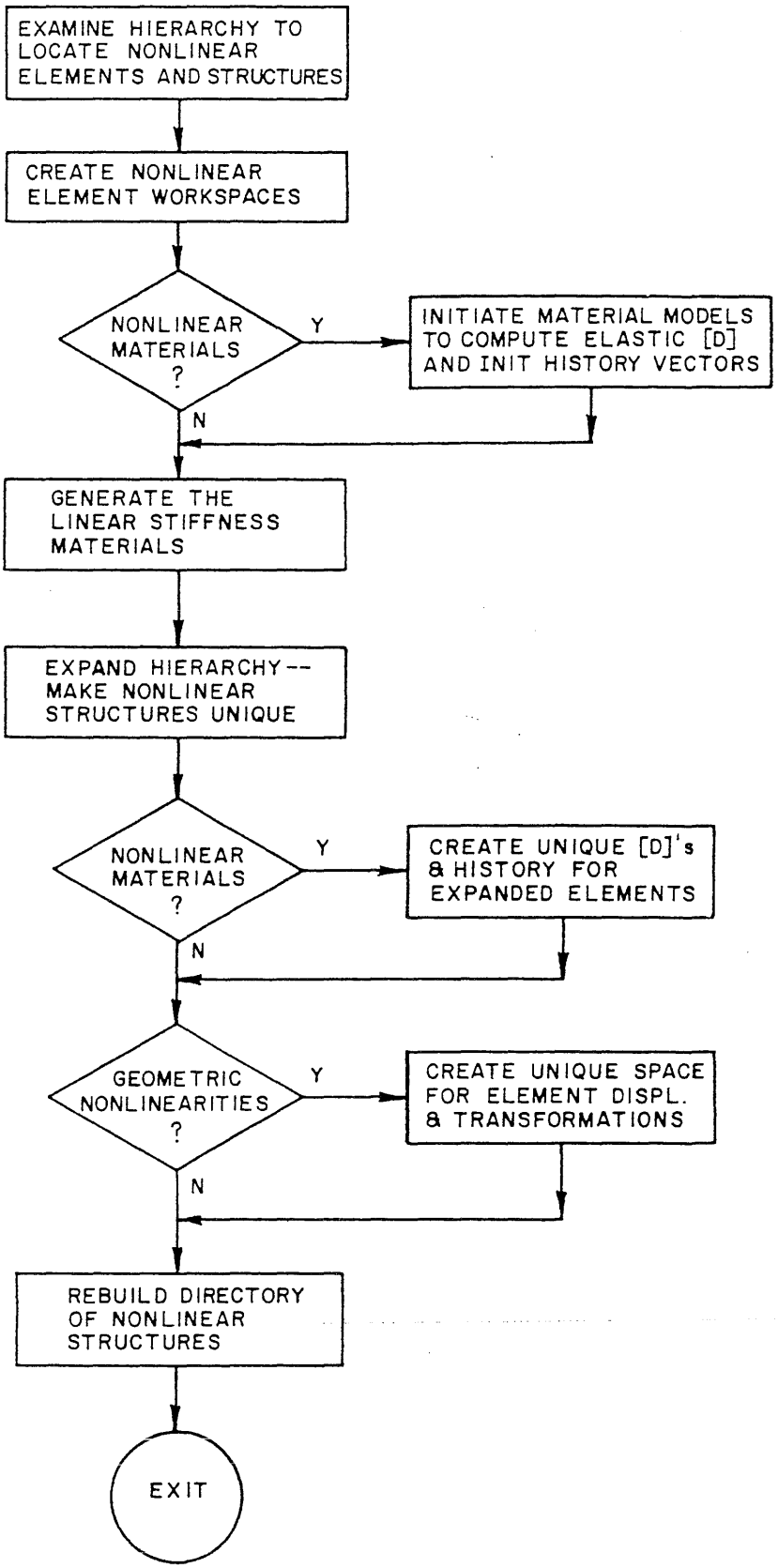


Fig. 5.6.2. Initialization of Nonlinear Data Structure

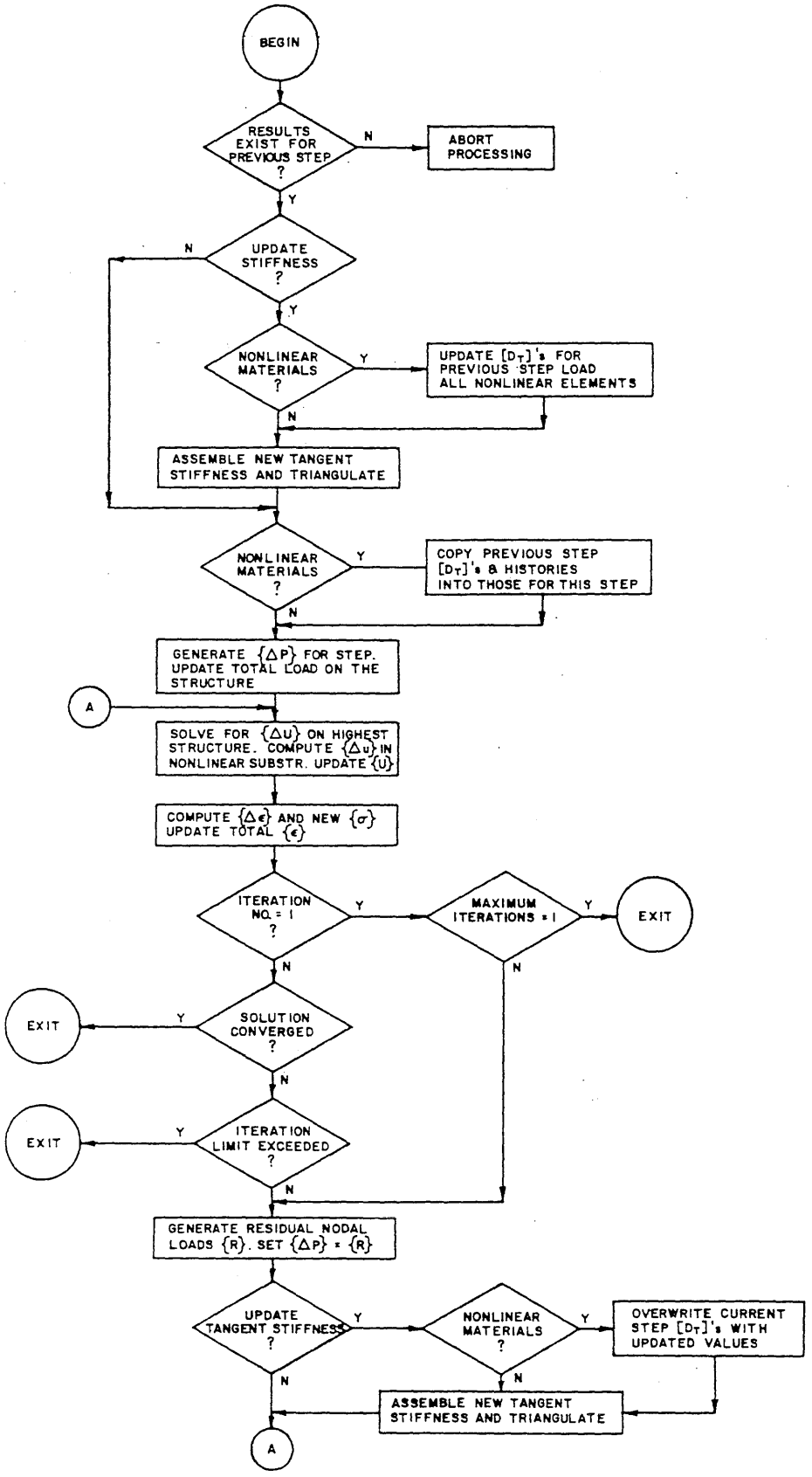


Fig. 5.6.3. Solution for a Load Step

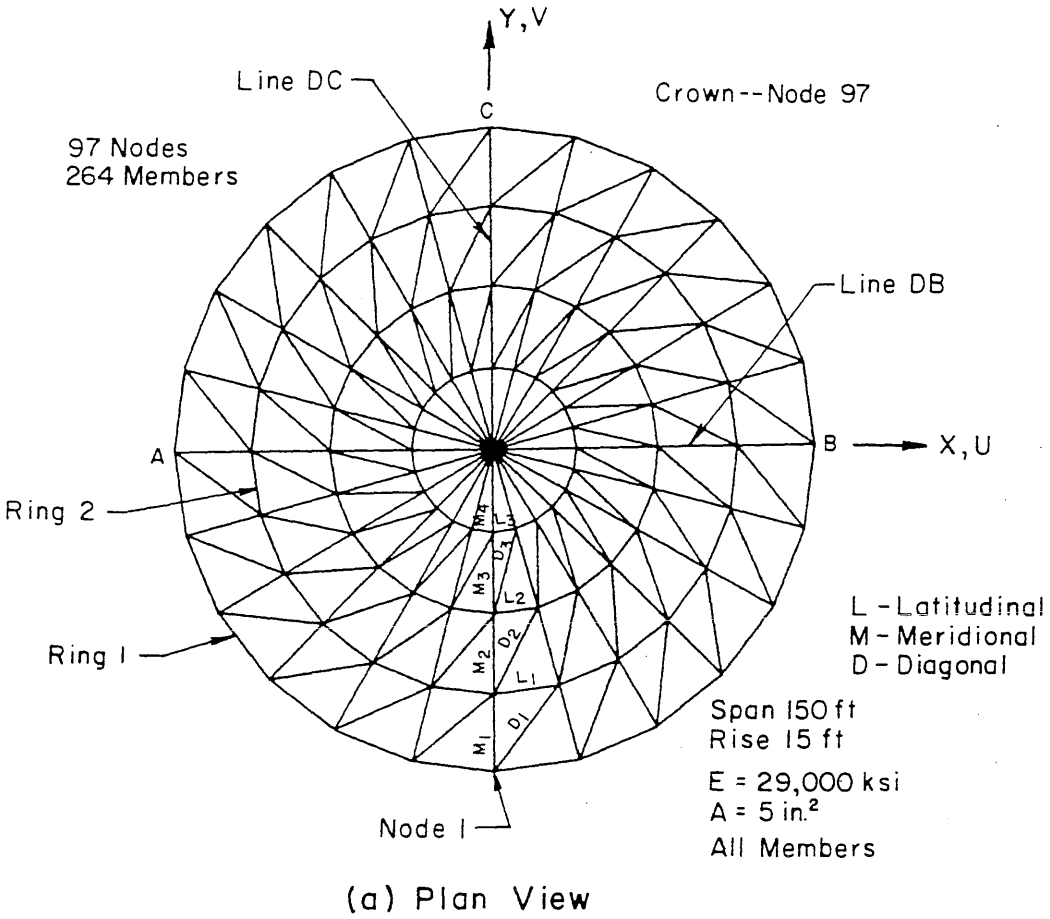
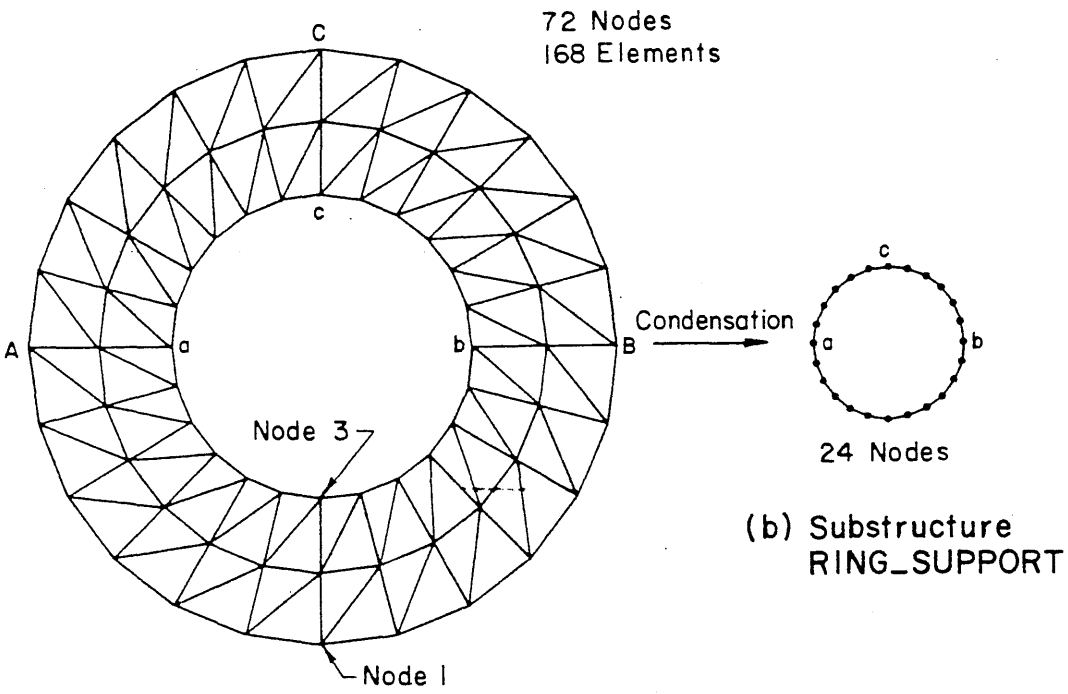
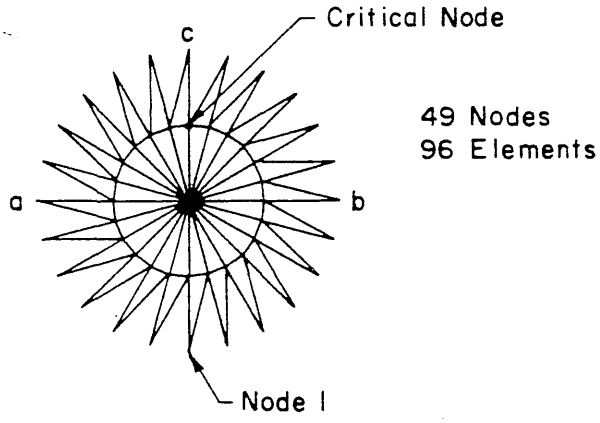


Fig. 6.2.1. Schwedler Dome



(a) Substructure BOTTOM_RING
(Linear)

(b) Substructure
RING_SUPPORT



(c) Structure DOME
(Nonlinear)

Fig. 6.2.2. Schwedler Dome - Substructured Model

```
C
   *RUN FINITE
C
C
C              NONLINEAR ANALYSIS OF A SHALLOW
C           SCHWEDLER DOME UNDER FULL UNIFORM LOADING
C
C           FINITE ELEMENT MODEL WITHOUT SUBSTRUCTURING
C
  STRUCTURE SHALLOWDOME
   NUMBER OF NODES 97 ELEMENTS 264
   ELEMENTS
     ALL TYPE SPACETRUSS GEOMETRICALLY NONLINEAR E 29000. AX 5.0

   COORDINATES
    1       0.0          -900.          0.0
    2       0.0          -682.77       78.17
    3       0.0          -458.91      134.55
            .
            .
            .
   97       0.0            0.0        180.
   INCIDENCES
    GENERATE 1-93 BY 4 FROM 1 2 ADD 4
    GENERATE 2-94 BY 4 FROM 2 3 ADD 4
    GENERATE 3-95 BY 4 FROM 3 4 ADD 4
    GENERATE 4-96 BY 4 FROM 4 97 ADD 4 0
    GENERATE 97-119 FROM 1 5 ADD 4
    120 93 1
    GENERATE 121-143 FROM 2 6 ADD 4
    144 94 2
    GENERATE 145-167 FROM 3 7 ADD 4
    168 95 3
    GENERATE 169-191 FROM 4 8 ADD 4
    192 96 4
    GENERATE 193-215 FROM 5 2 ADD 4
    216 1 94
    GENERATE 217-239 FROM 6 3 ADD 4
    240 2 95
    GENERATE 241-263 FROM 7 4 ADD 4
    264 3 96
C
C          DEFINE A UNIFORM ROOF LOADING EQUIVALENT TO 1 PSF.
C
   LOADING UNIT-ROOF
    NODAL LOADS
       2-94 BY 4 FORCE Z -0.2830
       3-95 BY 4 FORCE Z -0.19017
       4-96 BY 4 FORCE Z -0.0955
         97      FORCE Z -0.2870
C
C          DEFINE THE INCREMENTAL STEP LOADS
C
   LOADING FULL-ROOF
    NONLINEAR
      STEP 1 'TOTAL LOAD 30 PSF' COMBINE UNIT-ROOF 30.
      STEP 2 'TOTAL LOAD 60 PSF' COMBINE UNIT-ROOF 30.
C
   CONSTRAINTS
   1-93 BY 4 ALL = 0.
C
C
     MAXIMUM ITERATIONS 5
     CONVERGENCE TEST NORM RESIDUAL LOADS TOLERANCE 1.0
     TRACE NONLINEAR SOLUTION
     CONTINUE IF NONCONVERGENT
     UPDATE TANGENT STIFFNESS AT ITERATIONS 2 5
C
 COMPUTE NONLIN DISPL STRUCTURE SHALLOWDOME LOADING FULL-ROOF,
                STEP 1-2
  OUTPUT WIDE NONLIN DISPL FOR STRUCT SHALLOWDOME LOAD FULL-ROOF,
       STEP 1-2
```

Fig. 6.2.3. FINITE Input Data for Standard Model of Schwedler

```
*RUN FINITE
C
C
  STRUCTURE BOTTOM-RING
   NUMBER OF NODES 72 ELEMENTS 168
   ELEMENTS
     ALL TYPE SPACETRUSS  E 29000. AX 5.0
   INCIDENCES
    GENERATE 2 X 24 Y FROM 1 2 ADD 1 X 3 Y
    GENERATE 3 X 23 Y AS 49-117 FROM 1 4 ADD 1 X 3 Y
    GENERATE 2 X 23 Y AS 121-166 FROM 2 4 ADD 1 X 3 Y
    118  70 1
    119  71 2
    120  72 3
    167  71 1
    168  72 2
   COORDINATES
    1       0.0          -900          0.0
    2       0.0          -682.77       78.17
    3       0.0          -458.91       134.55
            .
            .
            .
   72     118.77         -443.27       134.56

C
 CONSTRAINTS
  1-70 BY 3 ALL = 0.0
C
C    NODAL LOADS CORRESPONDING TO 1 PSF FULL ROOF LOAD
C
 LOADING BOTTOM-RING
  NODAL LOADS
        3-72 BY 3 FORCE Z -0.19017
        2-71 BY 3 FORCE Z -0.2830
C
C
C        CONDENSED BOTTOM RING TRUSS
C
C
 STRUCTURE RING-SUPPORT
  NUMBER OF NODES 24 ELEMENTS 1
  ELEMENT 1 TYPE BOTTOM-RING CONDENSED
  INCIDENCES
   1  3-72 BY 3
  LOADING UNIT-ROOF
     EXTERNAL ELEMENT LOADS
       1  BOTTOM-RING
C
C
C        DEFINE THE FINAL DOME STRUCTURE
C
C
 STRUCTURE DOME
   NUMBER OF NODES 49 ELEMENTS 97
   ELEMENTS
    1-96 TYPE SPACETRUSS LARGE DISPLACEMENTS E 29000. AX 5.0
      97 TYPE RING-SUPPORT ROTATION SUPPRESSED
   INCIDENCES
     GENERATE 1-24 FROM 1 2 ADD 2
     GENERATE 25-48 FROM 2 49 ADD 2 0
     GENERATE 49-71 FROM 2 4 ADD 2
     GENERATE 73-95 FROM 2 3 ADD 2
     72  48 2
     96  48 1
     97  1-47 BY 2
   COORDINATES
   1       0.0          -458.91          134.55
   2       0.0          -230.57          168.61
   3    -118.77         -443.27          134.55
           .
           .
           .
  49       0.0             0.0           180.0
C
C          DEFINE A UNIFORM ROOF LOAD EQUIVALENT TO 1 PSF.
C
   LOADING UNIT-ROOF
    NODAL LOADS
       2-48 BY 2 FORCE Z -0.09550
         49      FORCE Z -0.28700
    EXTERNAL ELEMENT LOADS
       97 UNIT-ROOF 1.0
C
C          DEFINE THE INCREMENTAL STEP LOADS
C
  LOADING FULL-ROOF
   NONLINEAR
     STEP 1 'TOTAL LOAD 30 PSF' COMBINE UNIT-ROOF 30.0
     STEP 2 'TOTAL LOAD 60 PSF' COMBINE UNIT-ROOF 30.0
C
C
     MAXIMUM ITERATIONS 5
     CONVERGENCE TEST NORM RESIDUAL LOADS TOLERANCE 1.0
     TRACE NONLINEAR SOLUTION
     UPDATE TANGENT STIFFNESS AT ITERATIONS 2 5
C
 COMPUTE NONLIN DISPL STRUCTURE DOME LOADING FULL-ROOF,
                STEP 1-2
 OUTPUT WIDE NONLIN DISPL STRUCTURE DOME LOAD FULL-ROOF STEP 1-2
C
C          REQUEST COMPUTATION AND OUTPUT OF DISPLACEMENTS
C          AND STRESSES INSIDE THE CONDENSED LINEAR SUBSTRUCTURE
C
 OUTPUT WIDE NONLIN DISPL STRESSES FOR STRUCTURE DOME/97/1
        LOADING FULL-ROOF STEPS 1-2
 STOP
```

Fig. 6.2.4. FINITE Input Data for Substructured Model of Schwedler Dome

Ring Number (1)	DISPLACEMENT, IN INCHES					
	Linear Analysis			Nonlinear Analysis		
	U (2)	V (3)	W (4)	U (5)	V (6)	W (7)
2	-0.4671	0.2239	-1.782	-0.5251	0.2364	-1.9832
3	-0.3354	0.3881	-1.735	0.3400	0.4024	-1.7248
4	-0.1750	0.3755	-1.132	-0.1900	0.3838	-1.1895
5(crown)	0	0	1.941	0	0	1.8684

(a) Nodal Displacements Along DB

Member Series (1)	MEMBER STRESSES, IN KSI	
	Linear (2)	Nonlinear (3)
M_1	-20.58	-21.14
M_2	-14.63	-14.58
M_3	- 8.75	- 8.63
M_4	- 2.91	- 2.29
D_1	0	0.04
D_2	0	0.03
D_3	0	0.03
L_1	-19.84	-22.29
L_2	-21.19	-21.46
L_3	-22.01	-23.85

Note: Tension Positive

(b) Member Stresses

Fig. 6.2.5. Schwedler Dome -- Uniform Load Results

LINE DC Ring Number (1)	DISPLACEMENT, IN INCHES					
	STANDARD MODEL			SUBSTRUCTURED MODEL		
	U (2)	V (3)	W (4)	U (5)	V (6)	W (7)
2	-.120	-.392	-1.301	-.117	-.378	-1.248
3	-.218	-.395	-1.501	-.213	-.395	-1.500
4	-.246	-.516	-2.506	-.243	-.508	-2.452
5 (Crown)	-.134	-.349	0.464	-.131	-.345	0.458
LINE DB						
2	-.038	-.094	-.107	-.050	-.092	-.146
3	-.164	-.172	-.587	-.154	-.167	-.542
4	-.208	-.290	-.823	-.204	-.288	-.809

Fig. 6.2.6. Schwedler Dome -- Partial Loading Results for 29 PSF

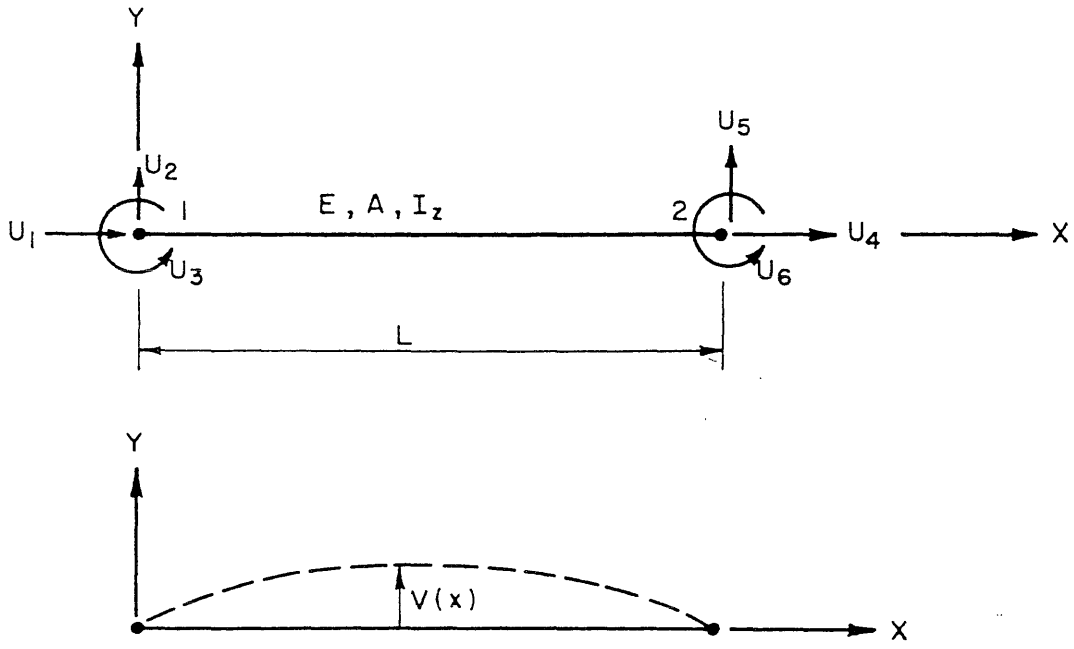


Fig. 6.3.1. Nonlinear Planeframe Element

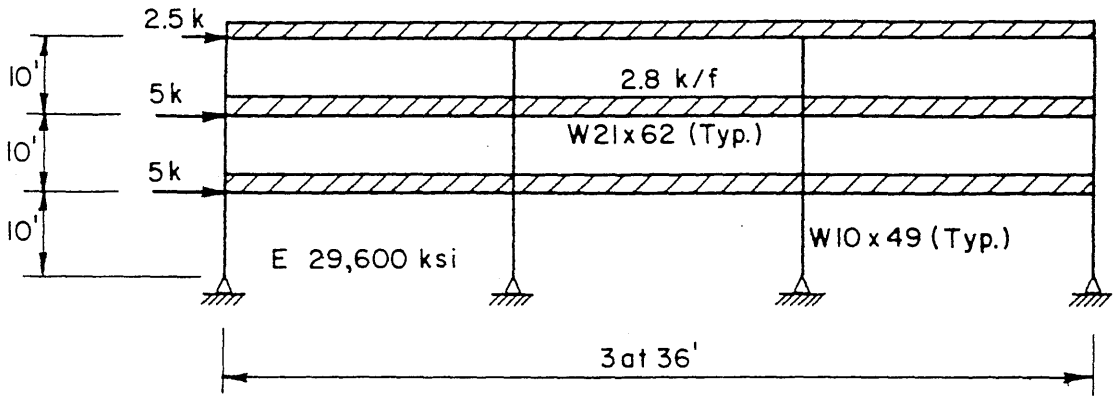


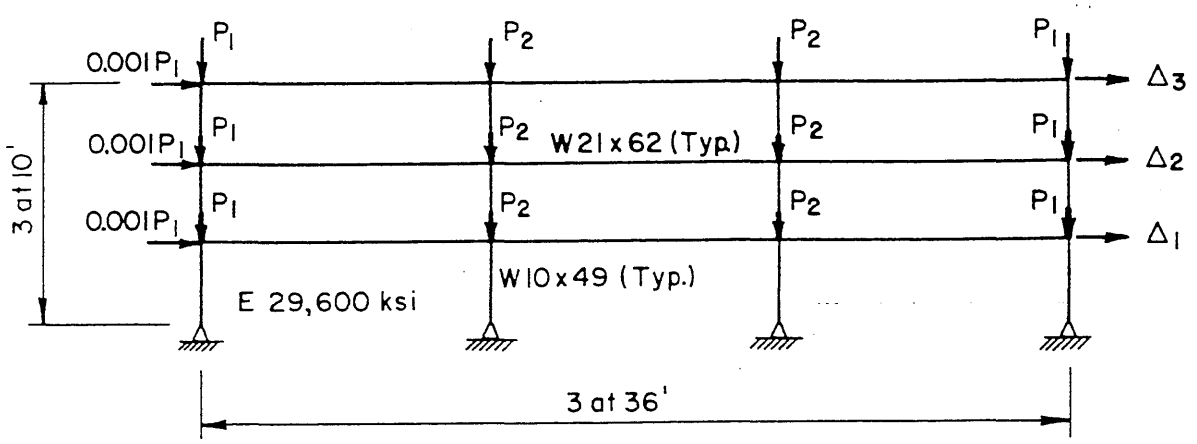
Fig. 6.3.2. Frame Geometry and Loading for P-Delta Analysis

DISPLACEMENTS (IN.)			
Floor Level	Linear Analysis	Ficticious Load	Finite
4	0.411	0.549	0.585
3	0.385	0.521	0.554
2	0.301	0.421	0.447
1	0	0	0

Fig. 6.3.3. Lateral Deflections for P-Delta Analysis

```
*RUN FINITE
C
C
C              P-DELTA ANALYSIS OF SIMPLE PLANE FRAME STRUCTURE
C
C
 ELEMENT BEAM  $ IGNORE AXIAL DEFORMATION IN BEAMS, I.E., AX = 500.
    TYPE PLANEFRAME E 29600.  TABLE W W21X62 AX 500.
    COORDINATES
      2  432
    LOADING FLOOR ' 1.0 KIPS PER INCH '
      ELEMENT LOADS
       UNIFORM FORCE Y W -1.0
C
 ELEMENT COLUMN
    TYPE PLANEFRAME LARGE DISPLACEMENTS E 29600.  TABLE W W10X49
    COORDINATES
       2   120
C
 STRUCTURE FRAME
   NUMBER OF NODES 16 ELEMENTS 21
   ELEMENTS
      1-12 TYPE COLUMN ROTATION Z 90.
      13-21 TYPE BEAM ROTATION SUPPRESSED
   INCIDENCES
      GENERATE 3 IN X 4 IN Y FROM 1 5 ADD 4 IN X 1 IN Y
      GENERATE 3 IN X 3 IN Y AS 13-21 FROM 5 6 ADD 1 IN X 4 IN Y
   CONSTRAINTS
      1-4 U V = 0.0
   LOADING WORKING 'BEAMS 2.8 K/F PLUS WIND'
      EXTERNAL ELEMENT LOADS
        13-21 FLOOR 0.23333
      NODAL LOADS
        5 9 FORCE X 5.0
        13  FORCE X 2.5
   LOADING COLUMN-LOADS 'EQUIVALENT COLUMN LOADS FROM GIRDERS'
      NODAL LOADS
        5 9 13 8 12 16 FORCE Y -50.4
        6 7 10 11 14 15 FORCE Y -100.8
   LOADING P-DELTA 'LOADING FOR NONLINEAR ANALYSIS'
      NONLINEAR
        STEP 1 'WORKING LOADS' WORKING 1.0
        STEP 2 'WORKING + 0.7*AXIAL 'COLUMN-LOADS 0.7
C
 CONVERGENCE TEST NORM RESIDUAL LOADS TOLERANCE 0.1
 TRACE NONLINEAR SOLUTION
 MAXIMUM ITERATIONS 10
 CONTINUE IF NONCONVERGENT
 PRINT RESIDUAL LOADS
 UPDATE STIFFNESS EVERY 2 ITERATIONS
C
 COMPUTE NONLINEAR DISPLACEMENTS FOR STRUCTURE FRAME,
         LOADING P-DELTA STEPS 1 2
 OUTPUT WIDE NONLINEAR DISPLACEMENTS STRESSES FOR STRUCTURE FRAME,
         LOADING P-DELTA STEPS 1 2
 STOP
```

Fig. 6.3.4. FINITE Input Data for P-Delta Analysis



For $\lambda = 1$, $P_1 = 1.0$ k

$P_2 = 2.0$ k

Fig. 6.3.5. Pattern Loads for Buckling Analysis of Planeframe Structure

λ	Floor Level	Linear Δ_L (in)	Nonlinear Δ_{NL} (in)	$\lambda_{cr} \frac{\lambda \Delta_{NL}}{\Delta_{NL} \Delta_L}$
200	4	0.0174	0.0642	274
	3	0.0161	0.0624	269
	2	0.0124	0.0537	260
			Average λ_{cr} =	267
250	4	0.0217	0.623	259
	3	0.0201	0.615	258
	2	0.0155	0.548	257
			Average λ_{ce} =	258
255	4	0.0221	2.24	257
	3	0.0205	2.22	257
	2	0.0158	1.98	257
			Average λ_{cr} =	257

Fig. 6.3.6. Summary of Displacements for Buckling Analysis of Planeframe Structure

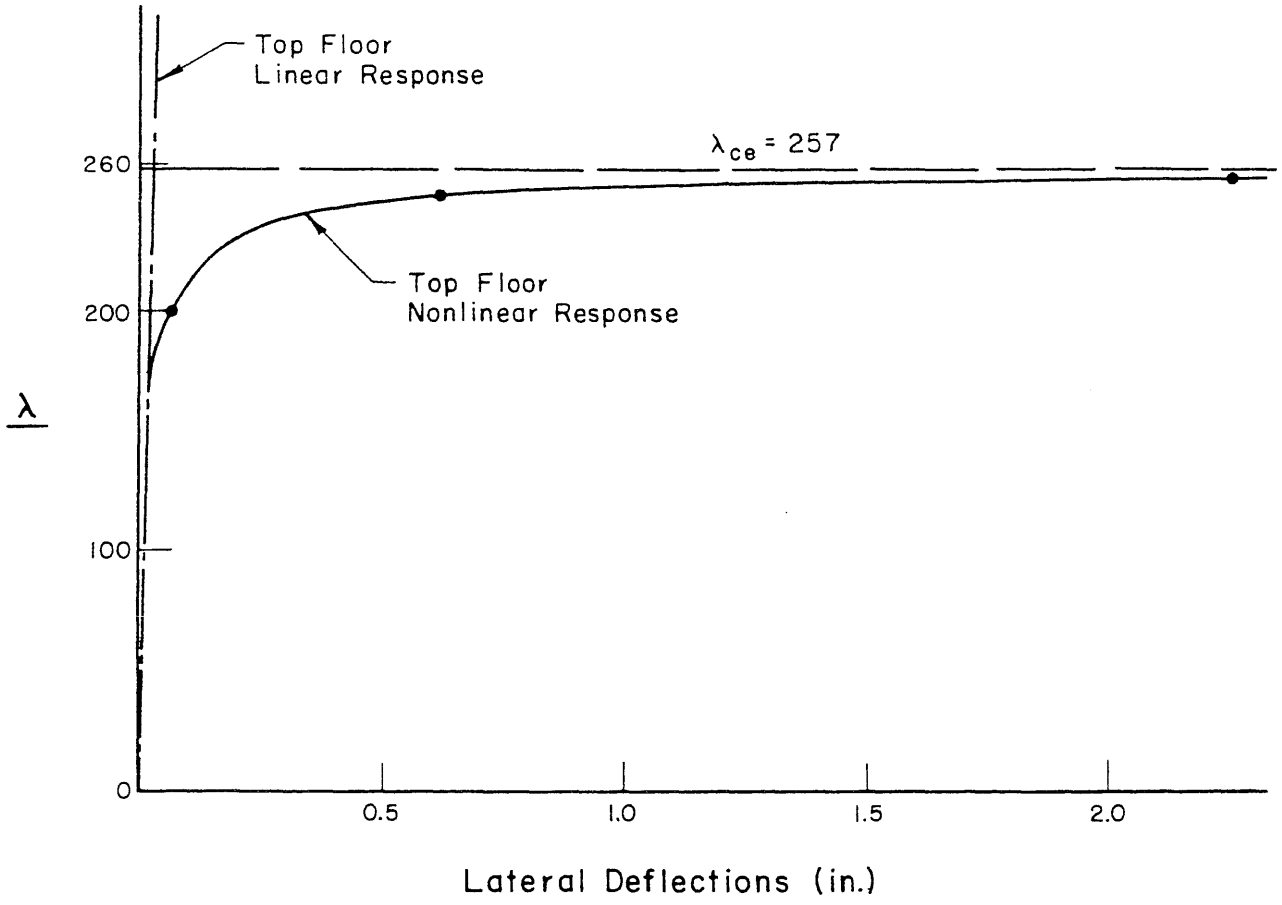


Fig. 6.3.7. Load-Deflection Plot for Planeframe Buckling Analysis

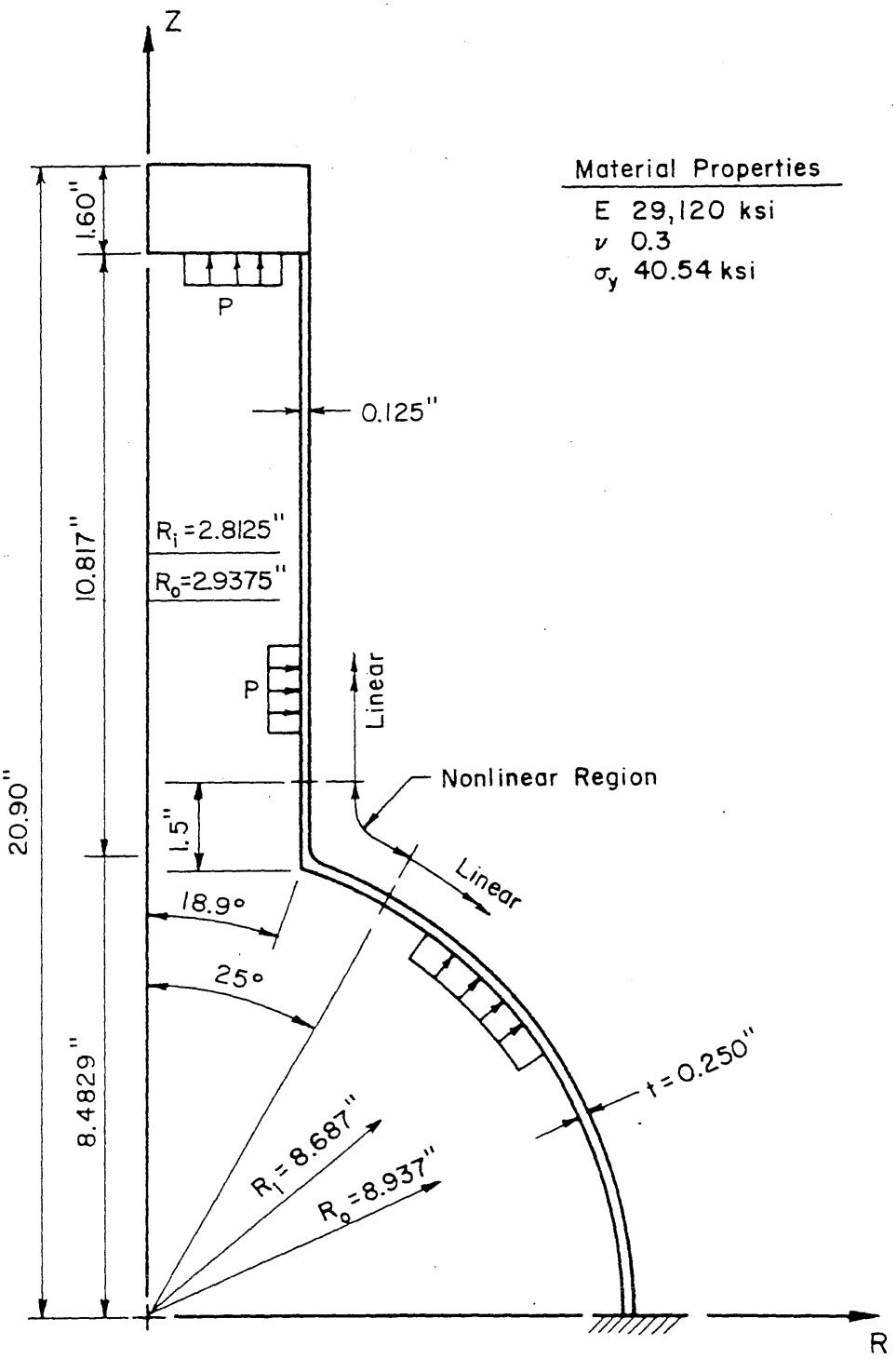
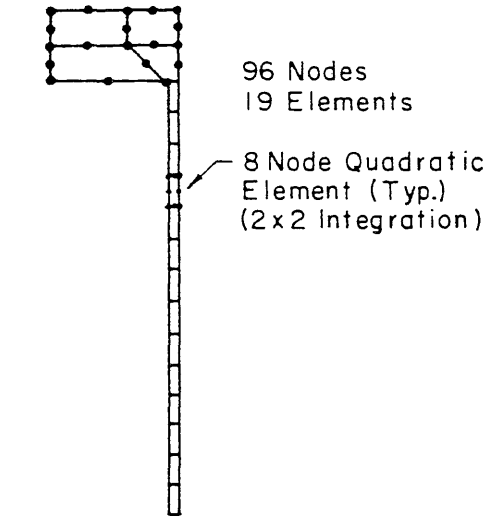
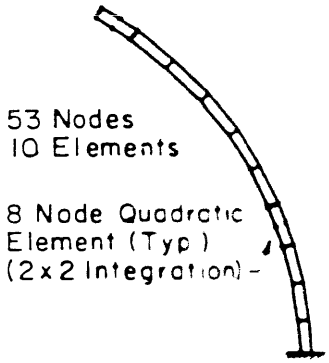


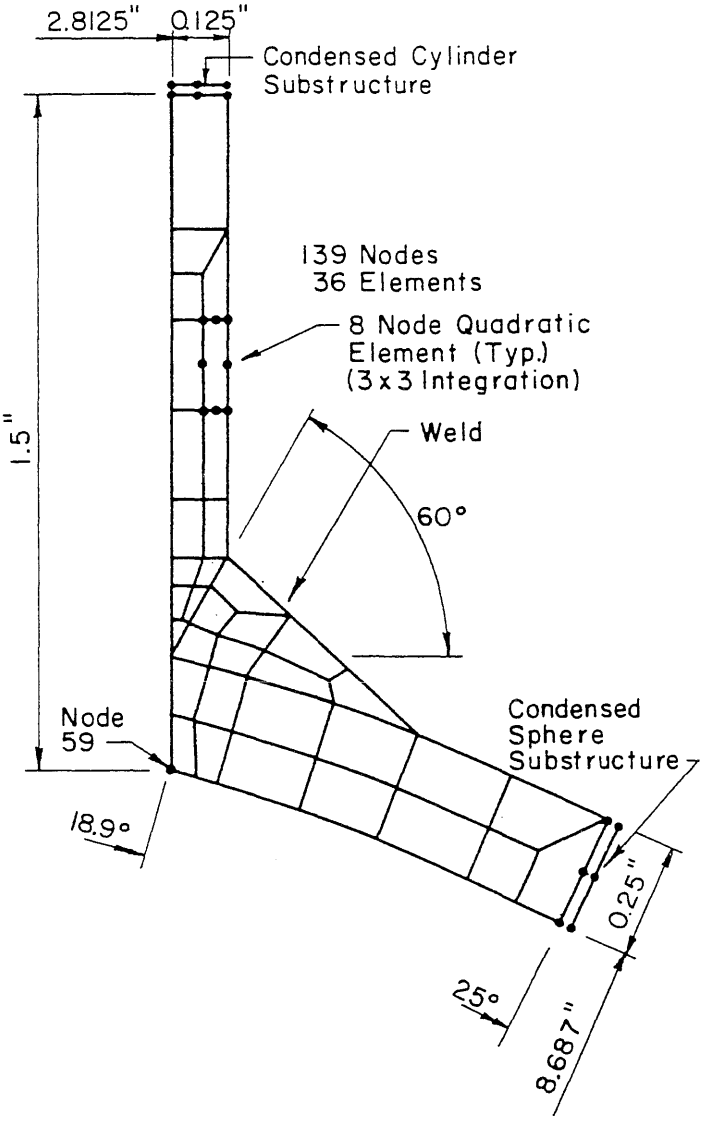
Fig. 6.4.1. Axisymmetric Pressure Vessel



(a) Substructure
CYL_AND_PLUG



(b) Substructure SPHERE1



(c) Structure JUNCTION

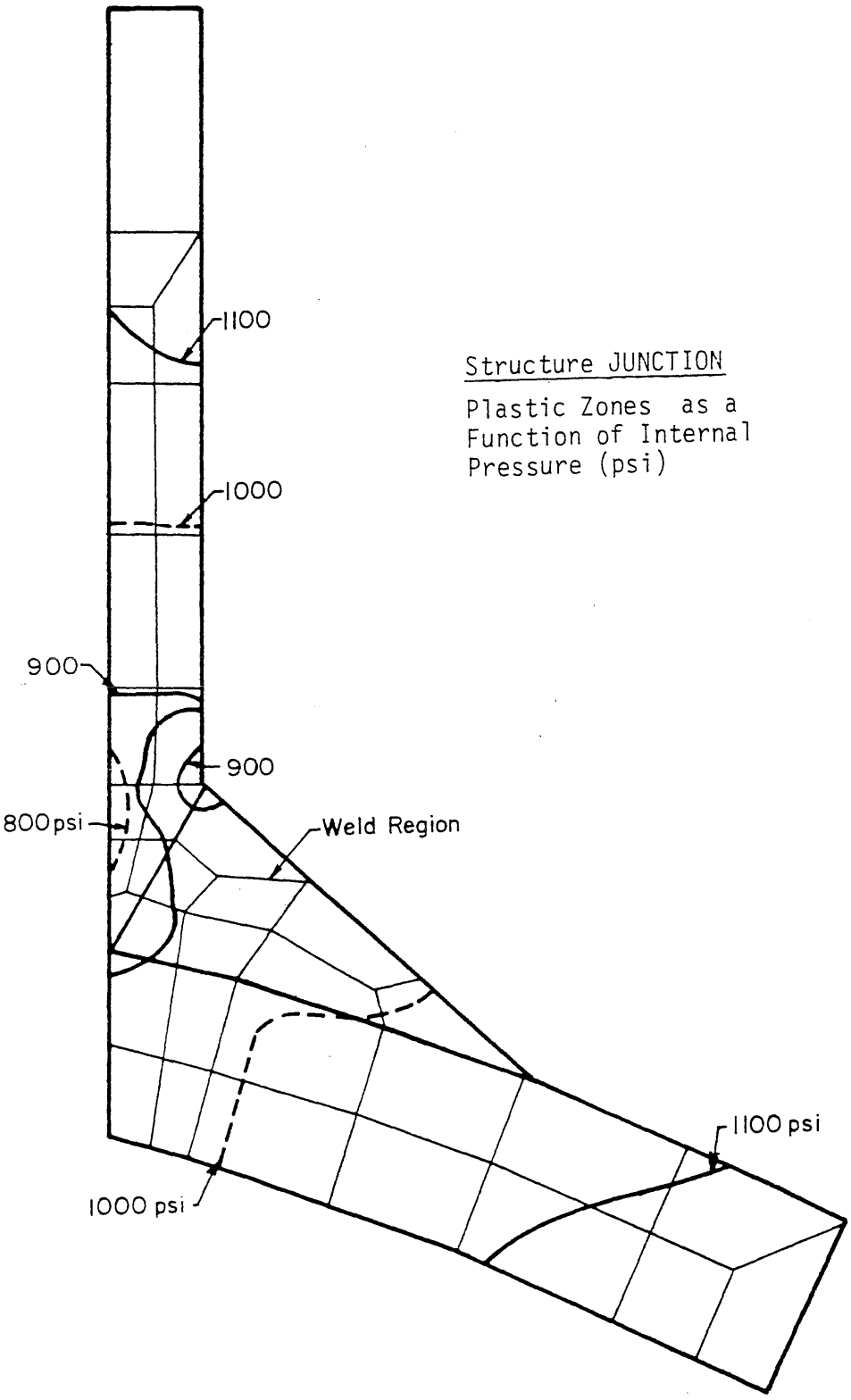
Fig. 6.4.2. Axisymmetric Pressure Vessel Finite Element Model

```
   *RUN FINITE
C
C          ONE ELEMENT OF CYLINDER
C
STRUCTURE CYLINDER-PIECE
   NUMBER OF NODES 8 ELEMENTS 1 COORDINATE POINTS 10
   ELEMENT 1 TYPE AXIQ2DISOP E 29000. NU 0.3 LINEAR NIX 2 NIY 2,
                              STRAINPTS 'NODPTS'
   INCIDENCES
      1 1-10
   COORDINATES
      1  2.8125  0.
      2  2.9375  0.0
      3  2.9375  0.6378
      4  2.8125  0.6378
      10 0.      10.
   LOADING UNIT-PRESSURE
      ELEMENT LOADS FOR TYPE AXIQ2DISOP
         1 UNIFORM FORCE X W 1.0 FACE 2
C
C
C              PLUG AND CYLINDER
C
STRUCTURE CYL-AND-PLUG
   NUMBER OF NODES 96 ELEMENT 19 COORDINATE POINTS 98
      ELEMENTS
         1-4 TYPE AXIQ2DISOP E 29000. NU 0.3 LINEAR NIX 2 NIY 2
         5-19 TYPE CYLINDER-PIECE ROTATION SUPPRESSED
      INCIDENCES
         1  5  13  11   3   4  12   8   7  97  98
         2  3  11   9   1   2  10   7   6  97  98
         3 13  21  19  11  12  20  16  15  97  98
         4 11  19  17   9  10  18  15  14  97  98
         5 24  26  21  13  22  23  25  16
         GENERATE 6-19 FROM 29 31 26 24 27 28,
            30 25  ADD 5 5 5 5 5 5,
            5 5
      COORDINATES
         1  0.0  20.90
         5  0.0  19.30
         3  0.0  20.10
         9  2.2031  20.90
         11 2.2031  20.10
         13 2.8125  19.30
         17 2.9375  20.90
         19 2.9375  20.10
         21 2.9375  19.30
         98 0.0   100.0
      CONSTRAINTS
         1-5  U = 0.0
      LOADING UNIT-PRESSURE
         ELEMENT LOADS FOR TYPE AXIQ2DISOP
            1 UNIFORM FORCE Y W 1.0 FACE 4
         EXTERNAL ELEMENT LOADS
            5-19 UNIT-PRESSURE
C
C          CONDENSED CYLINDER AND PLUG FOR NONLINEAR ANALYSIS.
C
STRUCTURE CYLINDER
   NUMBER OF NODES 3 ELEMENTS 1
   ELEMENT 1 TYPE CYL-AND-PLUG CONDENSED
   INCIDENCES
      1  94-96
   LOADING UNIT-PRESSURE
      EXTERNAL ELEMENT LOADS
         1  UNIT-PRESSURE
C
C        SPHERICAL VESSEL
C
STRUCTURE SPHERE1
   NUMBER OF NODES 53 ELEMENTS 10 COORDINATE POINTS 55
   ELEMENTS ALL TYPE AXIQ2DISOP E 29000. NU 0.3 NIX 2 NIY 2,
                              STRAINPTS 'NODPTS'
   INCIDENCES
      GENERATE 1-10 FROM 51 46 48 53 52 47 49 50 54 55,
               SUBTRACT  5 5 5 5 5 5 5 5 0 0
      COORDINATES
         1  8.687  0.
         3  8.937  0.0
         51 R 8.687 T 65.
         53 R 8.937 T 65.
         55  0.  100.
         GENERATE 1 IN R 10 IN THETA USING PATTERN 1-3 THETA 4 0 5,
                  THETA 6-8 POLAR
      CONSTRAINTS
         1-3 U V = 0.0
      LOADING UNIT-PRESSURE
         ELEMENT LOADS FOR TYPE AXIQ2DISOP
            1-10 UNIFORM NORMAL W 1.0 FACE 4
C
C          CONDENSED SPHERE FOR USE IN NONLINEAR ANALYSIS.
C
STRUCTURE SPHERE
   NUMBER OF NODES 3 ELEMENTS 1
   ELEMENT 1 TYPE SPHERE1 CONDENSED
   INCIDENCES
      1  51-53
   LOADING UNIT-PRESSURE
      EXTERNAL ELEMENT LOADS
         1  UNIT-PRESSURE
C
C
C             MATERIAL MODEL FOR VESSEL ELEMENTS IN NONLINEAR REGION
C
C
 MATERIAL STEEL TYPE VON-MISES
   PROPERTIES TOLERANCE 0.001 ALPHA 0.05 MAXINCR 30
   USE STRESS-STRAIN FUNCTION SEGMENTAL
     PROPERTIES
         YMODULUS 29120. NU 0.3 ELASTO-PLASTIC,
         TENSYIELD 40.54
```

Fig. 6.4.3. FINITE Input Data for Axisymmetric Pressure Vessel Analysis

```
C
C       JUNCTION OF SPHERE AND CYLINDER.
C
 STRUCTURE JUNCTION
   NUMBER OF NODES 139 ELEMENTS 38 COORDINATE POINTS 141
   ELEMENTS
     1-36 TYPE AXIQ2DISOP MATERIAL STEEL NIX 2 NIY 2
     37   TYPE CYLINDER ROTATION SUPPRESSED
     38   TYPE SPHERE   ROTATION SUPPRESSED
   INCIDENCES
      1 6 8 3 1 4 5 7 2                  140 141
      2 11 13 8 6 9 10 12 7              140 141
      3 17 19 13 11 15 16 18 12          140 141
                .
                .
                .
   COORDINATES
        ORIGIN X 2.8125 Y 8.2329
        1   0.  1.5
            .
            .
        GENERATE 2 IN X 2 IN Y USING PATTERN 17-19 Y 22 0 23,
                 Y 25-27
            .
            .
            .
   LOADING UNIT-PRESS 'INTERNAL PRESSURE 100 PSI'
     ELEMENT LOADS FOR TYPE AXIQ2DISOP
      1 2 3 5 7 9 11 13 15 UNIFORM NORMAL W 0.1 FACE 2
      24-36 BY 2 UNIFORM NORMAL W 0.1 FACE 4
     EXTERNAL ELEMENT LOADS
      37 38 UNIT-PRESSURE 0.1
   LOADING PRESSURE 'NONLINEAR LOADING CONDITION'
     NONLINEAR
        STEP 1 '700 PSI INTERNAL PRESSURE' COMBINE UNIT-PRESS 7.
        STEP 2 '800 PSI INTERNAL PRESSURE' UNIT-PRESS 1.0
        STEP 3 '900 PSI INTERNAL PRESSURE' UNIT-PRESS 1.0
        STEP 4 '1000 PSI INTERNAL PRESSURE' UNIT-PRESS 1.0
        STEP 5 '1100 PSI INTERNAL PRESSURE' UNIT-PRESSURE 1.0
C
C
   MAXIMUM ITERATIONS 1
   CONVERGENCE TEST NORM RESIDUAL LOADS TOLERANCE 1.0
   TERMINATE IF NONCONVERGENT
   UPDATE TANGENT STIFFNESS AT ITERATION 2
   TRACE NONLINEAR SOLUTION
C
C
 COMPUTE NONLINEAR DISPLACEMENTS FOR STRUCTURE JUNCTION,
         FOR LOADING PRESSURE STEP 1
 OUTPUT WIDE NONLINEAR DISPLACEMENTS ALL STRESSES 1-36,
         FOR STRUCTURE JUNCTION LOADING PRESSURE STEP 1
 STOP
```

Fig. 6.4.3. continued



Structure JUNCTION

Plastic Zones as a Function of Internal Pressure (psi)

Fig. 6.4.4. Spread of Plastic Zones for Axisymmetric Pressure Vessel

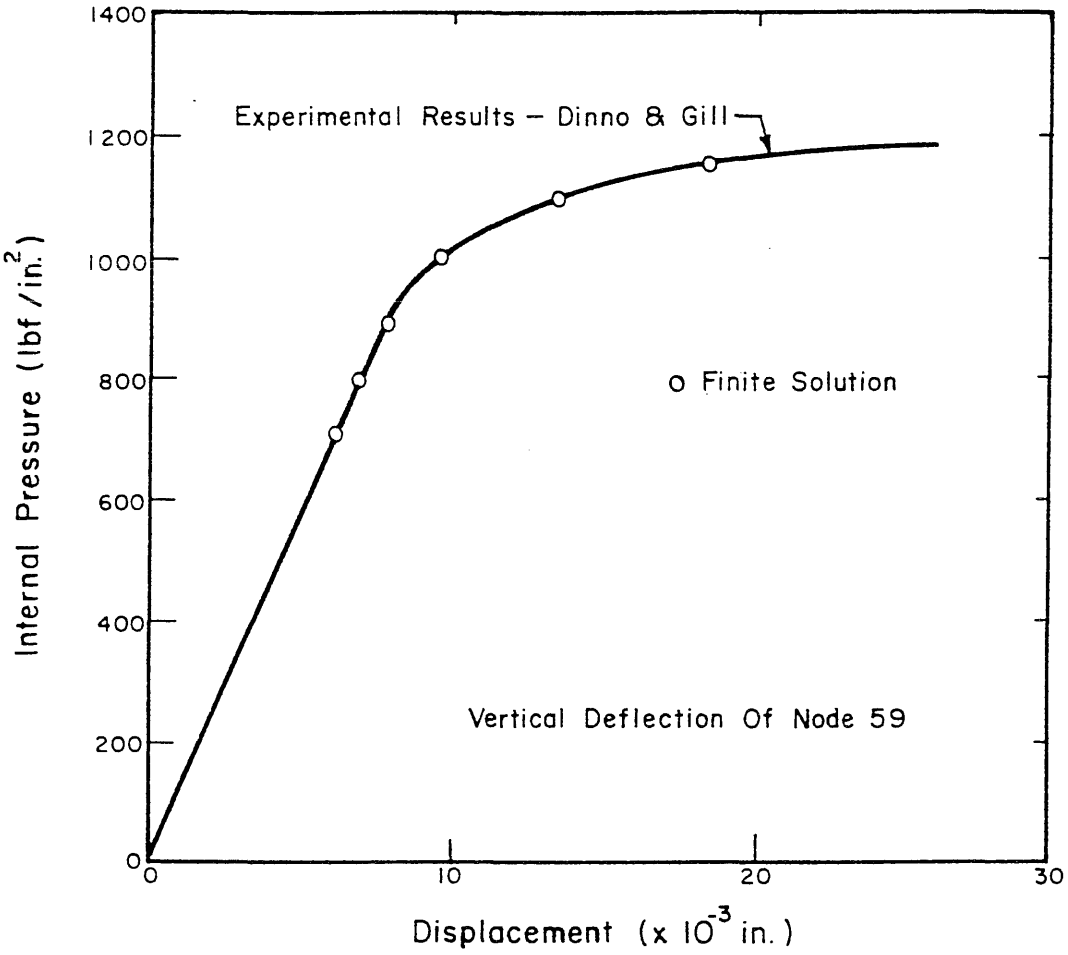
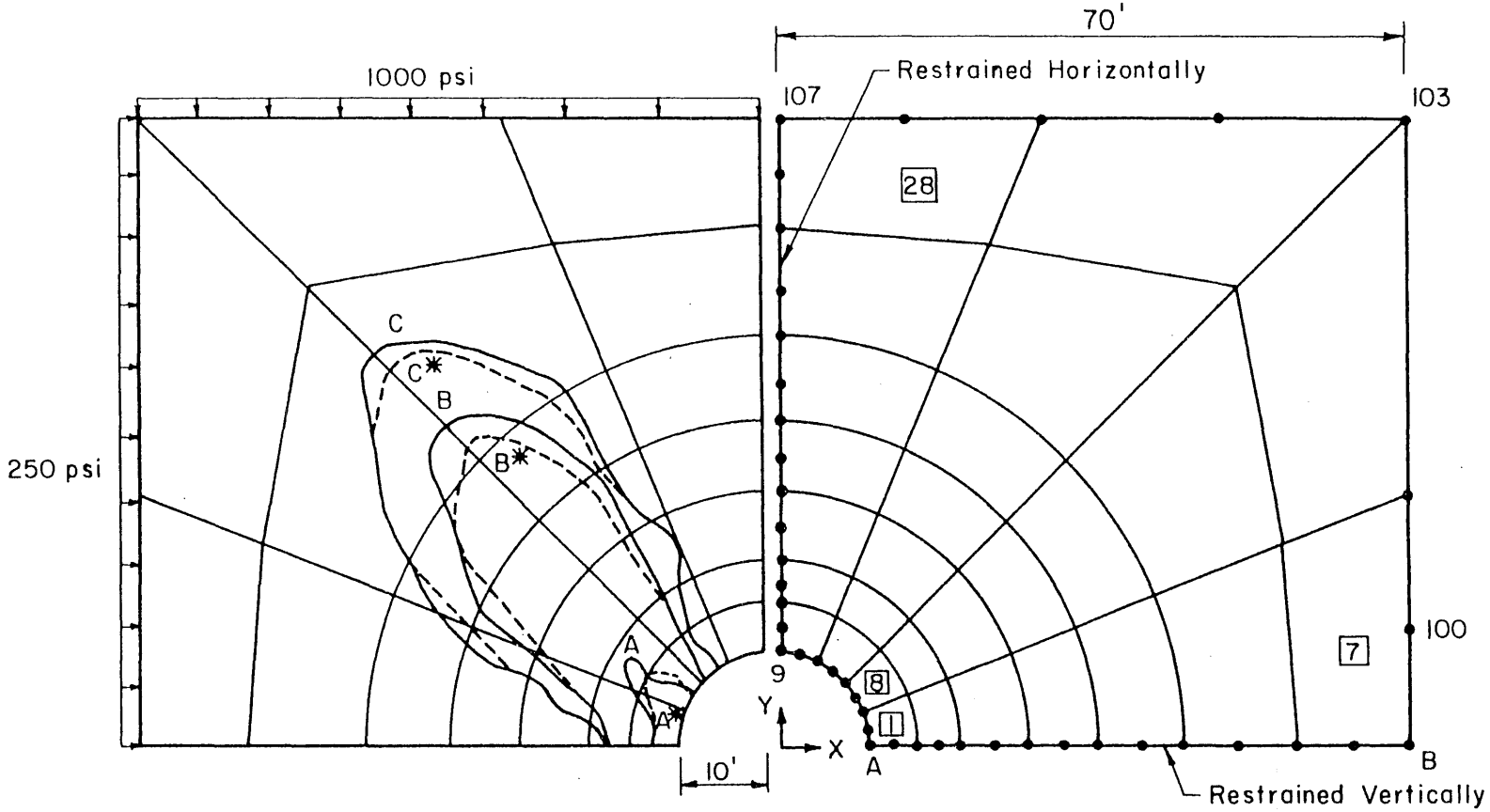


Fig. 6.4.5. Axisymmetric Pressure Vessel Displacements



(a) Plastic Zones for Gradual Excavation

(b) Finite Element Mesh

Legend

% Tunnel Excavated

FINITE	Reyes & Deere
A – 25%	A* – 23%
B – 75%	B* – 75%
C – 100%	C* – 100%

Material Properties

E = 500 ksi ν = 0.2
Cohesion = 0.28 ksi
Internal Friction = 30°

Fig. 6.5.1. Tunnel Excavation in Deep Rock

```
*RUN FINITE
C
C
C               NONLINEAR ANALYSIS OF TUNNEL EXCAVATION
C               ---------------------------------------
C
C         GRADUAL REMOVAL OF SUPPORTS INSIDE A CIRCULAR SHAPED
C         TUNNEL IN A ROCKY TYPE MATERIAL.  THE DRUCKER-PRAGER
C         YIELD CRITERION, ASSOCIATED FLOW RULE, AND PERFECT
C         PLASTICITY ARE EMPLOYED TO MODEL THE ROCK MATERIAL.
C
C
C
  MATERIAL ROCK TYPE DRUCKER-PRAGER
     PROPERTIES  E 500000. NU 0.2 COHESION 280. PHEE 30.,
                 MAXDFSTRESS 5.0 PSTRAIN
C
C
 STRUCTURE TUNNEL
C
    NUMBER OF NODES 107 ELEMENTS 28
    ELEMENTS ALL TYPE Q2DISOP MATERIAL ROCK THICKNESS 1.0
C
    INCIDENCES
     GENERATE 7 IN X 4 IN Y FROM 1 15 17 3 2 16 10 11,
              USING PATTERN 1 10 15 Y 2 0 16 Y 3 11 17
C
    COORDINATES
      1    120.    0.
      9      0.  120.
     10    150.    0.
             .
             .
             .
     GENERATE 1-9 POLAR
     GENERATE 10-14 POLAR
     GENERATE 15-23 POLAR
     GENERATE 24-28 POLAR
     GENERATE 29-37 POLAR
     GENERATE 38-42 POLAR
     GENERATE 43-51 POLAR
C
C
C               BOTTOM -- NO VERTICAL DISPLACEMENT
C               LEFT SIDE -- NO HORIZONTAL DISPLACEMENT
C
C
    CONSTRAINTS
     1-99 BY 14 V = 0.
     10-98 BY 14 V = 0.
     9-107 BY 14 U = 0.
     14-106 BY 14 U = 0.
C
C
C               INITIAL OVERBURDEN AND LATERAL PRESSURE
C               LOADINGS.  ABOVE 1000 PSI, FROM RIGHT 250 PSI.
C
C
C
    LOADING OVERBURDEN 'INITIAL BOUNDARY STRESSES'
     ELEMENT LOADS FOR TYPE Q2DISOP
       7, 14 UNIFORM NORMAL W -250. FACE 1
      21, 28 UNIFORM NORMAL W -1000. FACE 1
C
C
C               EQUIVALENT PRESSURES EXERTED ON TUNNEL FACE
C               THAT PRODUCE SAME INITIAL EFFECTS AS IF TUNNEL
C               DID NOT EXIST.
C
C
C
    LOADING TUNNEL-SUPPORT 'EQUIVALENT PRESSURE FOR NO TUNNEL'
     ELEMENT LOADS FOR TYPE Q2DISOP
      1 DISTRIBUTED GLOBAL FORCE Y W1 0. W2 195.1 W3 382.7 FACE 2
      8 DISTRIBUTED GLOBAL FORCE Y W1 382.7 W2 555.6 W3 707.1 FACE 2
     15 DISTRIBUTED GLOBAL FORCE Y W1 707.1 W2 831.5 W3 923.9 FACE 2
     22 DISTRIBUTED GLOBAL FORCE Y W1 923.9 W2 980.8 W3 1000. FACE 2
      1 DISTRIBUTED GLOBAL FORCE X W1 250. W2 249. W3 231. FACE 2
      8 DISTRIBUTED GLOBAL FORCE X W1 231. W2 207.9 W3 176.8 FACE 2
     15 DISTRIBUTED GLOBAL FORCE X W1 176.8 W2 138.9 W3 95.7 FACE 2
     22 DISTRIBUTED GLOBAL FORCE X W1 95.7 W2 48.8 W3 0. FACE 2
C
C
C               NONLINEAR LOADING CONDITION WITH STEPS
C               DEFINED TO SIMULATE GRADUAL EXCAVATION OF THE
C               TUNNEL.
C
C
C
    LOADING EXCAVATION
     NONLINEAR
       STEP 1 'INITIAL STRESSES NO TUNNEL' OVERBURDEN 1.0,
              TUNNEL-SUPPORT 1.0
       STEP 2 '25 % OF TUNNEL SUPPORT REMOVED' TUNNEL SUPPORT -0.25
       STEP 3 '50 % OF TUNNEL SUPPORT REMOVED' TUNNEL-SUPPORT -0.25
       STEP 4 '75 % OF TUNNEL SUPPORT REMOVED' TUNNEL-SUPPORT -0.25
       STEP 5 'TUNNEL COMPLETED -- NO SUPPORT' TUNNEL-SUPPORT -0.25
C
C
C               STRUCTURE AND LOADS DEFINITION COMPLETED.
C               SPECIFY PARAMETERS CONTROLLING NONLINEAR
C               SOLUTION.
C
C
C
    MAXIMUM ITERATIONS 1
    CONVERGENCE TEST NORM RESIDUAL LOADS TOLERANCE 0.1
    UPDATE TANGENT STIFFNESS AT ITERATIONS 3 6 9
    TRACE NONLINEAR SOLUTION
C
C
C
 COMPUTE NONLINEAR DISPLACEMENTS FOR STRUCTURE TUNNEL,
         LOADING EXCAVATION STEP 1-5
 OUTPUT WIDE NONLINEAR DISPLACEMENTS STRESSES FOR STRUCTURE TUNNEL,
         LOADING EXCAVATION STEP 1-5
  STOP
```

Fig. 6.5.2. FINITE Input Data for Excavation of Deep Tunnel in Rock

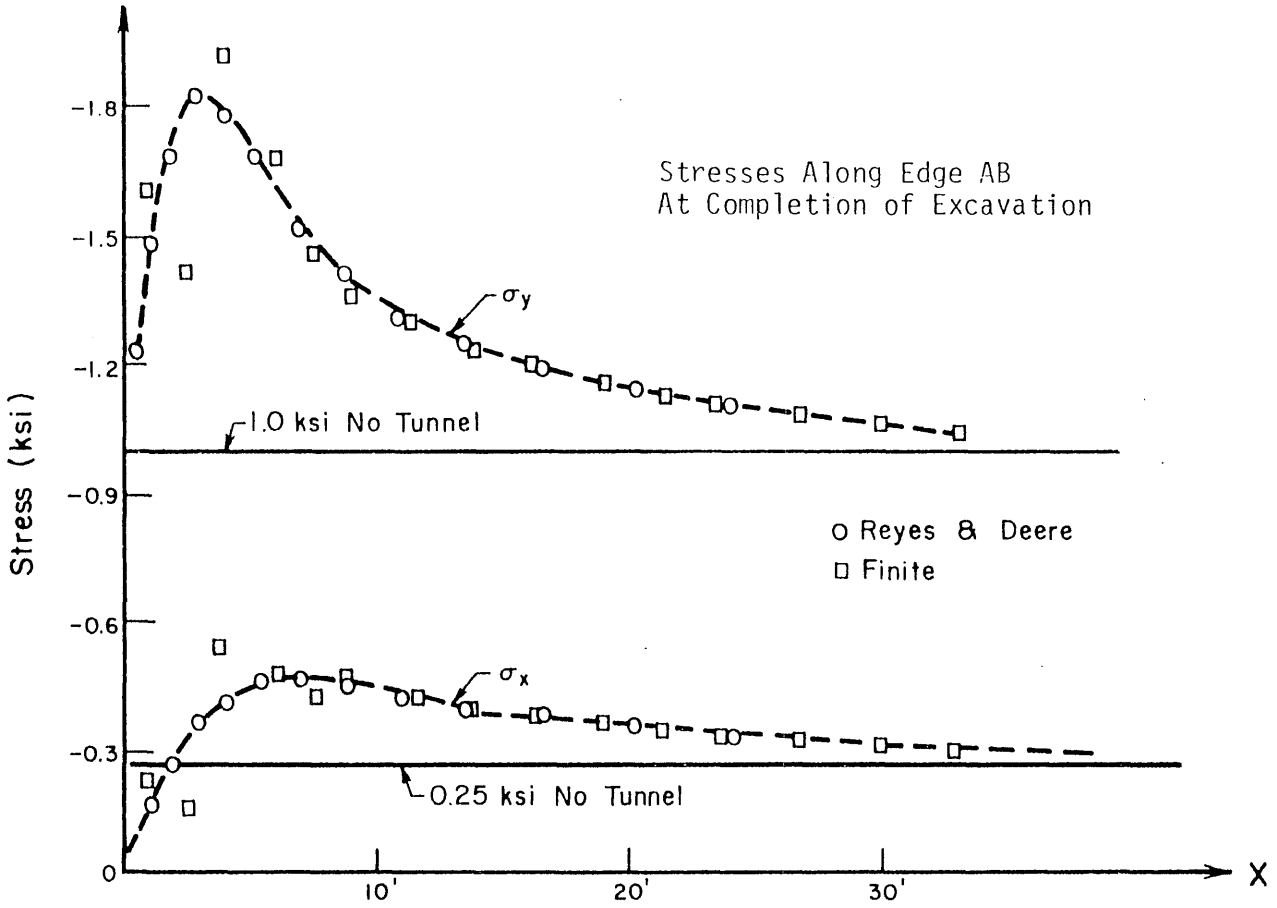


Fig. 6.5.3. Redistribution of Stresses in Deep Tunnel

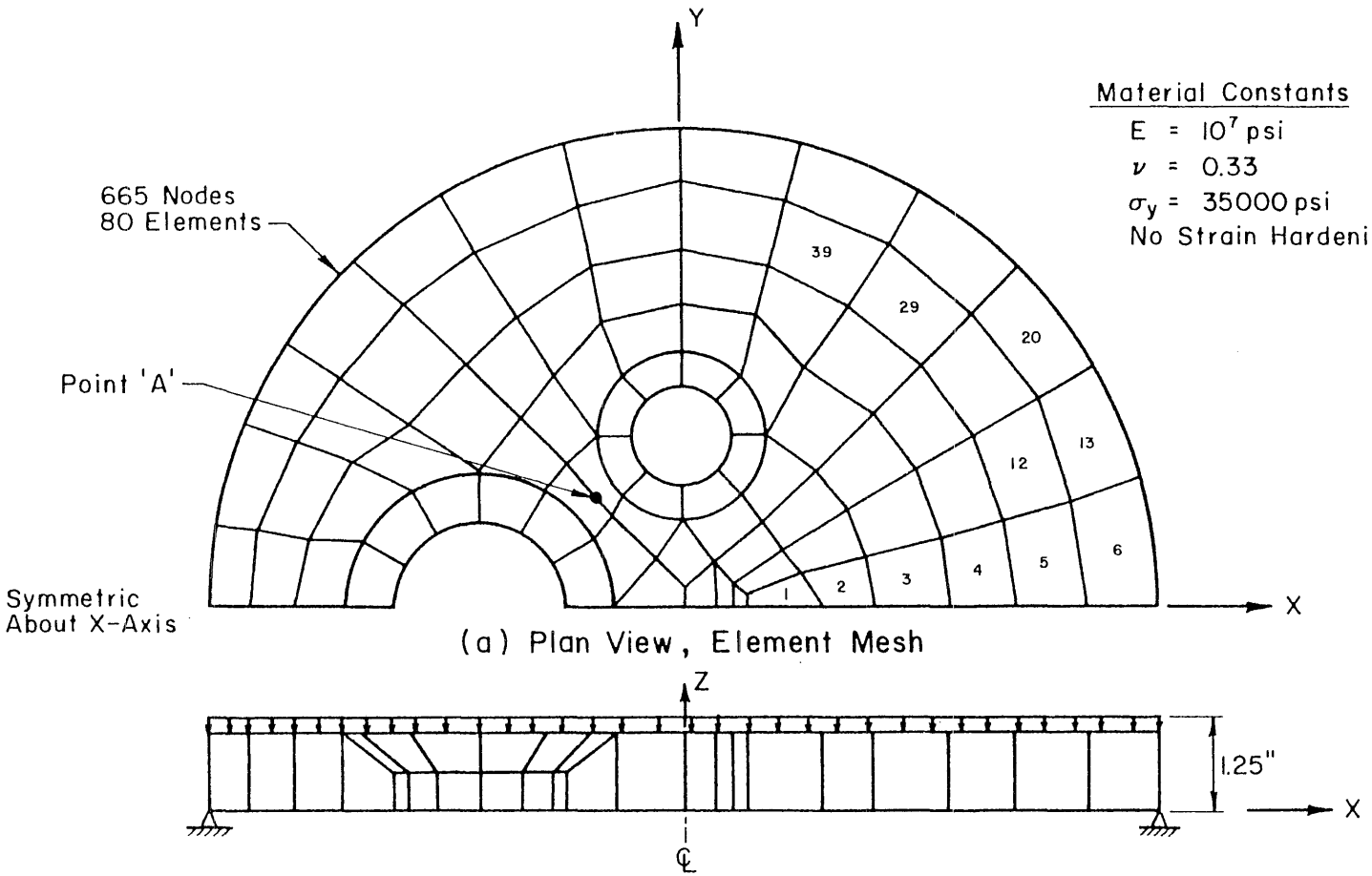


Fig. 6.6.1. Thick Penetrated Plate

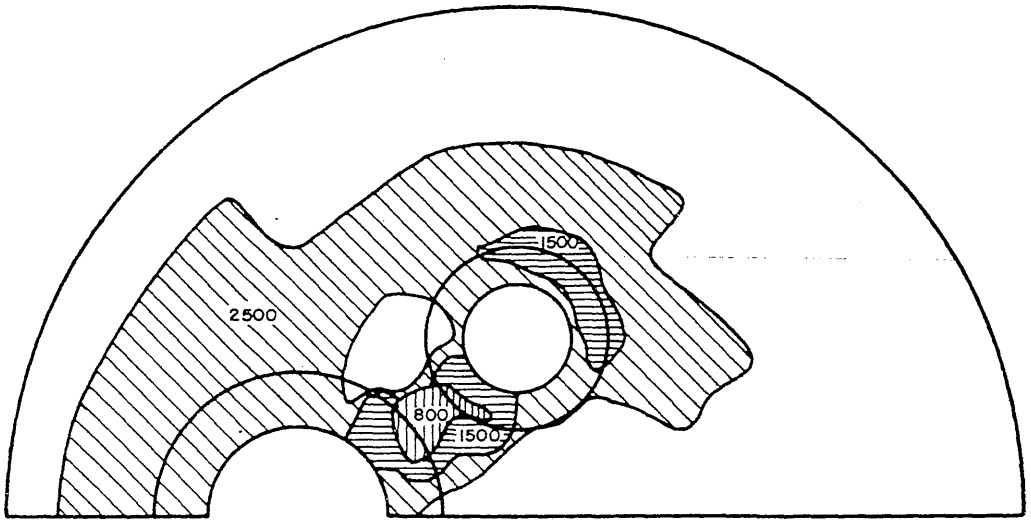
```
*RUN FINITE
C
C
C
C          ELASTO-PLASTIC 3-D ANALYSIS OF A CIRCULAR
C          PENETRATED THICK PLATE UNDER PRESSURE
C
C
C
C
C
C          MATERIAL PROPERTIES:
C
C               YOUNG'S MODULUS  -- 10. E06 PSI
C               POISSON'S RATIO  -- 0.33
C               YIELD STRESS     -- 35000 PSI
C
C
C
  MATERIAL ALUMINUM TYPE VON-MISES
    PROPERTIES TOLERANCE 0.001 ALPHA 0.05 MAXINCR 30
    USE STRESS-STRAIN FUNCTION SEGMENTAL
      PROPERTIES
          YMODULUS 10.E06 NU 0.33 ELASTO-PLASTIC TENSYIELD 35000.
C
C
  STRUCTURE PLATE
    NUMBER OF NODES 665 ELEMENTS 80
    ELEMENTS
C       LINEAR ELEMENTS
            3-6 10-13 18-20 28-30 39 40 48 49 58 59,
          65 66 72 73 77-80  TYPE Q3DISOP  E 10.E06 ,
          NU 0.33
C       NONLINEAR ELEMENTS
          1-2 7-9 14-17 21-27 31-38 41-47 50-57,
          60-64 67-71 74-76 TYPE Q3DISOP MATERIAL ALUMINUM NIZ 5
C
C
    COORDINATES
    1          0.90000000          0.00000000          0.00000000
    2          0.90000000          0.00000000          0.62500000
    3          0.90000000          0.00000000          1.25000000
                  .
                  .
    GENERATE 217   301   304
    GENERATE 302-304
    GENERATE 292   294   299
    GENERATE 297-299
   INCIDENCES
      1     3  15  20   8   1  13  18   6   2  14,
           19   7   5  17  16   4  10   9  11  12
      2    15  27  32  20  13  25  30  18  14  26,
           31  19  17  29  28  16  22  21  23  24
      3    27  39  44  32  25  37  42  30  26  38,
           43  31  29  41  40  28  34  33  35  36
      4    39  51  56  44  37  49  54  42  38  50,
                        .
                        .
                        .
```

```
           587 655 651 583 582 650 646 645 652 653
     79    656 588 595 663 654 586 593 661 655 587,
           594 662 658 590 589 657 653 652 659 660
     80    663 595 602 611 661 593 600 609 662 594,
           601 610 665 597 596 664 660 659 605 606
C
C
     CONSTRAINTS
C
C       SIMPLY SUPPORTED CIRCULAR EDGE.
C
         73 76 78 130 132 184 186 257 259 354,
         356 427 429 499 501 551 553 607 609,
         664 661 657 654 650 647 640 642 ALL = 0.
C
C       PLANE OF SYMMETRY.
C
         642-644  638  639  635-637  631  632  628-631,
         624  625  621-623  617 618  612-614  262-264,
         270  271  201-203  206 207  189-191  197 198,
         135-137  143 144  81-83  89 90  1-3  9 10 13-15,
         21 22 25-27  33 34 37-39  45 46 49-51  57  58,
         61-63  69  70  73-75  V = 0.
C
C
C
     LOADING UNIT-PRESSURE 'UNIFORM PRESSURE'
        ELEMENT LOADS FOR TYPE Q3DISOP
          ALL UNIFORM GLOBAL FORCE Z FACE 5 P -1.0
C
     LOADING PRESSURE
        NONLINEAR
          STEP 1 'TOTAL PRESSURE 100 PSI' COMBINE UNIT-PRESSURE 100
C
C
     MAXIMUM ITERATIONS 1
C
C
   COMPUTE NONLINEAR DISPLACEMENTS FOR STRUCTURE PLATE,
            LOADING PRESSURE STEP 1
   OUTPUT WIDE NONLINEAR DISPLACEMENTS STRESSES,
            FOR STRUCTURE PLATE LOADING PRESSURE STEP 1
C
C
  STOP
```

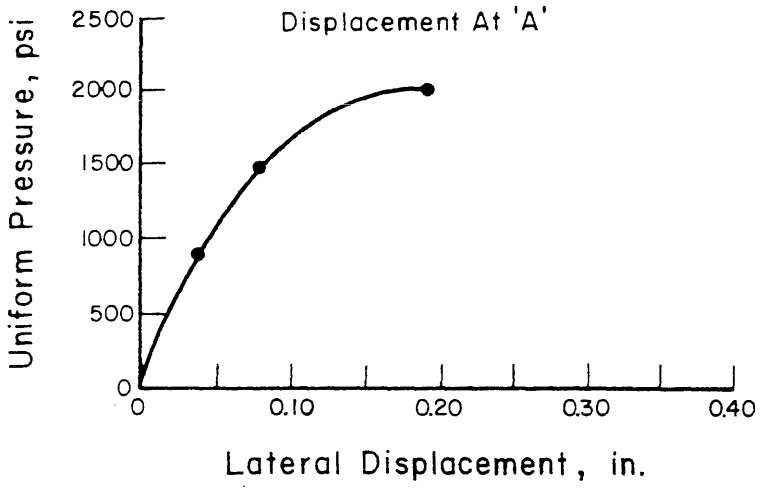
Fig. 6.6.2. FINITE Input Data for Penetrated Plate Structure

```
  *RUN FINITE FILE = 20, , 22
C
C          RESTART ANALYSIS OF THICK PENETRATED PLATE.
C          DEFINE THE 3 RD LOAD STEP, REQUEST TRACE OF THE
C          NONLINEAR SOLUTION PROCESS, RESET THE MAXIMUM NUMBER
C          OF ITERATIONS AND CONVERGENCE TEST, THEN REQUEST
C          SOLUTION FOR DISPLACEMENTS.
C          STRESSES ARE REQUESTED ONLY FOR THE NONLINEAR ELEMENTS.
C
  ACCESS STRUCTURE PLATE
C
    LOADING PRESSURE
       STEP 3 'TOTAL PRESSURE 1500 PSI' UNIT-PRESSURE 700.
C
  TRACE NONLINEAR SOLUTION
  MAXIMUM ITERATIONS 10
  CONVERGENCE TEST NORM RESIDUAL LOADS TOLERANCE 1.0
C
  DESTROY NONLINEAR RESULTS FOR STRUCTURE PLATE LOAD PRESSURE STEP 1
  COMPUTE NONLINEAR DISPLACEMENTS FOR STRUCTURE PLATE,
          FOR LOADING PRESSURE STEP 3
  OUTPUT WIDE NONLINEAR DISPLACEMENTS FOR STRUCTURE PLATE,
          LOADING PRESSURE STEP 3
   OUTPUT WIDE NONLINEAR STRESSES 1-2,
          7-9 14-17 21-27 31-38 41-47 50-57,
         60-64 67-71 74-76 FOR STRUCTURE PLATE,
         LOADING PRESSURE STEP 3
STOP
```

Fig. 6.6.3. FINITE Input Data for Penetrated Plate Analysis Restart



(a) Yielding At Top Layer Integration Points For Values of Applied Uniform Pressure



(b) Lateral Deflection At Point A

Fig. 6.6.4. Results for Thick Penetrated Plate

```
C
C
C          POL TABLE DATA TO DEFINE ELEMENT SPACETRUSS
C          ___________________________________________
C
C
 DEFINE ELEMENT SPACETRUSS
   NUMBER OF NODES 2
   NODAL FREEDOMS
     ALL U V W
   SUBROUTINE 1
   PROPERTIES
      E       REAL  1.0
      G       REAL  0.0
      NU      REAL  0.3
      ALPHA   REAL  0.0
      MASS     REAL  0.0
      DENSITY REAL  0.0
      DEBUG LOGICAL FALSE
      AX   REAL 0.0
   END OF PROPERTIES
   NUMBER OF STRAIN POINTS 2
   NUMBER OF REAL STRAINS 1
   WORKSPACE 7 WORDS
   NONLINEARITY GEOMETRIC MATERIAL
   STRAIN DECLARATIONS
      ALL  EPX - SIGMA_X, DELTAL - FORCE_X
   LOADS
     TYPE CONCENTRATED
       P INTENSITY REAL -1.0
       L REAL 0.0
       FRACTIONAL LOGICAL FALSE
     TYPE UNIFORM
          .
          .
          .
   END OF LOADS
 END OF ELEMENT
```

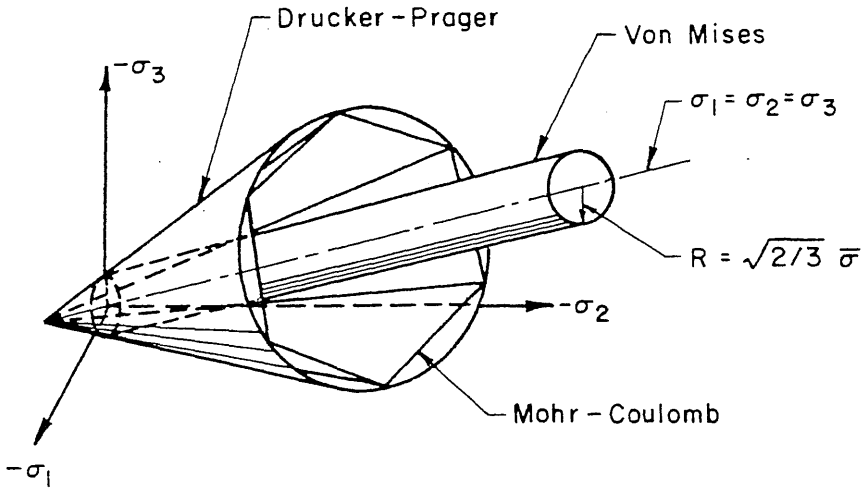
Fig. A.1. Spacetruss Element Definition Data for FINITE

```
C ***************************************************************
C *                                                             *
C *  STIFFNESS GENERATOR FOR ROD, PLANETRUSS, AND SPACETRUSS    *
C *                                                             *
C ***************************************************************
C
C
      SUBROUTINE ASG01( COORD, PROPS, K, M, N, IROW, JCOL, IERR,
     1                  WRKSPA, DMTRIX, STRESS, DISPL, NLTYPE )
C
C
C             STIFFNESS GENERATOR FOR THE ROD, PLANETRUSS, AND
C             SPACETRUSS ELEMENTS.  THE FORMULATION PERMITS
C             LINEAR, MATERIAL NONLINEAR, GEOMETRIC NONLINEAR,
C             AND COMBINED NONLINEAR BEHAVIOR.  ALL HIGHER
C             ORDER TERMS ( DERIVATIVES SQUARED ) ARE INCLUDED
C             IN GEOMETRIC NONLINEAR STIFFNESS.  THE NOTATION
C             FOLLOWS THAT OF NAYAK AND ZIENCKEWICZ.
C
C
C
      REAL  L, L2, K
      DIMENSION   COORD(3,1), PROPS(1), K(M,N), WRKSPA(1), DMTRIX(1),
     1            STRESS(1), DISPL(1)
      COMMON /ELECOM/  L, L2, IPPT, MATNON, GEONON,  DX, DY, DZ,
     1                 E, AX, NDOF, STIFF(6,6), DU, DV, DW, SIGMAX,
     2                 FACTOR, B(6), SUM, KK, J, I, LL, DEBUG, IOUT
      EQUIVALENCE ( RPPT,IPPT ), ( RBUG,DEBUG )
      LOGICAL  DEBUG, MATNON, GEONON
C
C
C             THE FIRST TIME FINITE CALLS THE ROUTINE COMPUTE
C             THE 6 X 6 STIFFNESS FOR A SPACETRUSS ELEMENT IN
C             THE ELEMENT COMMON AREA.  ON SUBSEQUENT CALLS
C             THE APPROPRIATE SUBMATRIX ( WITH CORRECT SIZE FOR
C             TYPE OF ELEMENT ) IS RETURNED.
C
C
      IF ( .NOT. ( IROW.EQ.1 .AND. JCOL.EQ.1 ) ) GO TO 100
C
      RPPT = PROPS(1)
      RBUG = PROPS(9)
      DX = COORD(1,2) - COORD(1,1)
      DY = COORD(2,2) - COORD(2,1)
      DZ = COORD(3,2) - COORD(3,1)
      L = SQRT( DX*DX + DY*DY + DZ*DZ )
      IF ( L .GT. 0.0 ) GO TO 1
      CALL ASEERR( IOUT )
      WRITE(IOUT,2) L
 2    FORMAT(6X,23HTHE COMPUTED LENGTH IS , F10.3,//)
      IERR = - 1
      RETURN
 1    CONTINUE
      MATNON = AND( 1,NLTYPE ) .NE. 0
      GEONON = AND( 2,NLTYPE ) .NE. 0
      GEONON = GEONON .AND. ( AND( 4,NLTYPE ) .EQ. 0 )
      IF ( NLTYPE .GT. 0 ) WRKSPA(1) = L
      E = PROPS(3)
      AX = PROPS(IPPT)
      NDOF = M
      DO 15 J = 1, 6
      DO 10 I = 1, 6
      STIFF(I,J) = 0.0
 10   CONTINUE
 15   CONTINUE
      DU = 0.0
      DV = 0.0
      DW = 0.0
      IF ( MATNON ) E = DMTRIX(1)
      IF ( .NOT.GEONON ) GO TO 30
      DU = DISPL(NDOF+1) - DISPL(1)
      IF ( NDOF .GE. 2 ) DV = DISPL(NDOF+2) - DISPL(2)
      IF ( NDOF .EQ. 3 ) DW = DISPL(NDOF+3) - DISPL(3)
      SIGMAX = STRESS(1)
C
C             COMPUTE INITIAL STRESS PART OF STIFFNESS FOR
C             GEOMETRIC NONLINEAR.
C
C               K(SIGMA) = TRANS(G) * M * G
C
C
      FACTOR = AX * SIGMAX / L
      DO 20 I = 1, 6
      STIFF(I,I) = FACTOR
 20   CONTINUE
      FACTOR = -FACTOR
      STIFF(4,1) = FACTOR
      STIFF(5,2) = FACTOR
      STIFF(6,3) = FACTOR
      STIFF(1,4) = FACTOR
      STIFF(2,5) = FACTOR
      STIFF(3,6) = FACTOR
C
C
C             ADD THE ORDINARY LINEAR STIFFNESS AND HIGHER
C             ORDER GEOMETRICALLY NONLINEAR TERMS.
C
C
C                 K = K(SIGMA) + TRANS(B) * E * B
C
```

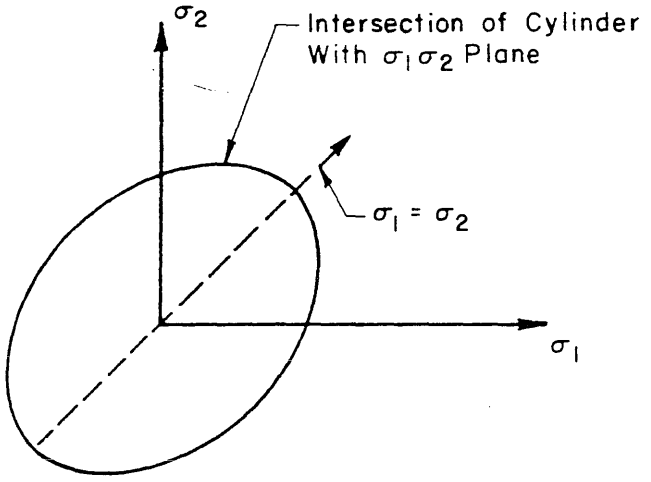
Fig. A.2. Spacetruss Stiffness Generation Subprogram

```
C
C
 30   CONTINUE
      L2 = L * L
      B(1) = - ( DU/L + 1. ) / L
      B(2) = -DV / L2
      B(3) = -DW / L2
      B(4) = -B(1)
      B(5) = -B(2)
      B(6) = -B(3)
      FACTOR = AX * E * L
      DO 50 I = 1, 6
      DO 40 J = 1, 6
      STIFF(I,J) = STIFF(I,J)  + ( B(I) * B(J) * FACTOR )
 40   CONTINUE
 50   CONTINUE
      IF ( .NOT. DEBUG ) GO TO 100
      CALL ASEERR( IOUT )
      IERR = 0
      WRITE(IOUT,1000) MATNON, GEONON, SIGMAX, DU, DV, DW, L
 1000 FORMAT(//,3X,13HTRUSS ELEMENT  ,/,3X,2L5,5F10.5 )
      WRITE(IOUT,1020) E, AX
 1020 FORMAT(3X,2F20.6)
      WRITE(IOUT,1010) ((STIFF(I,J),J=1,6),I=1,6)
 1010 FORMAT(/,3X,16HSTIFFNESS MATRIX,6(/,5X,6F10.3) )
C
C
C              PASS REQUESTED K SUBMATRIX BACK TO FINITE
C
C
 100  CONTINUE
      KK = ( IROW - 1 ) * 3
      LL =  ( JCOL - 1 ) * 3
      DO 120 I = 1, M
      DO 110 J = 1, M
      K(I,J) = STIFF(KK+I,J+LL)
 110  CONTINUE
 120  CONTINUE
C
      RETURN
      END
```

Fig. A.2. continued



(a) Yield Surfaces



(b) Von Mises Ellipse

Fig. B.1. Various Yield Surfaces

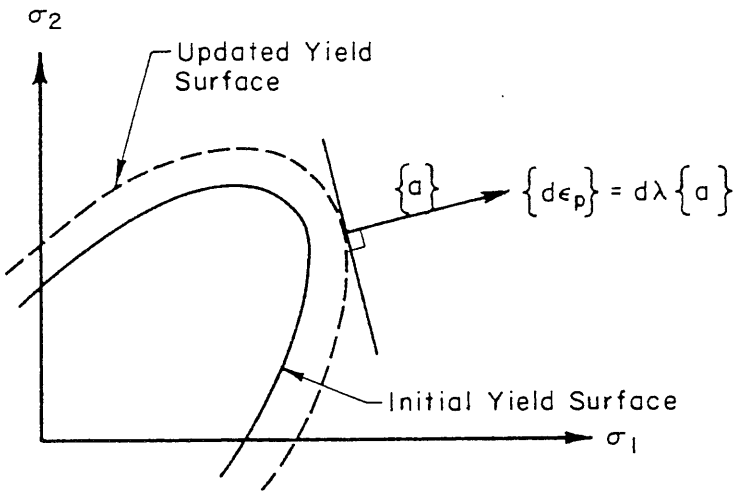


Fig. B.2. Normality Principle and Isotropic Hardening

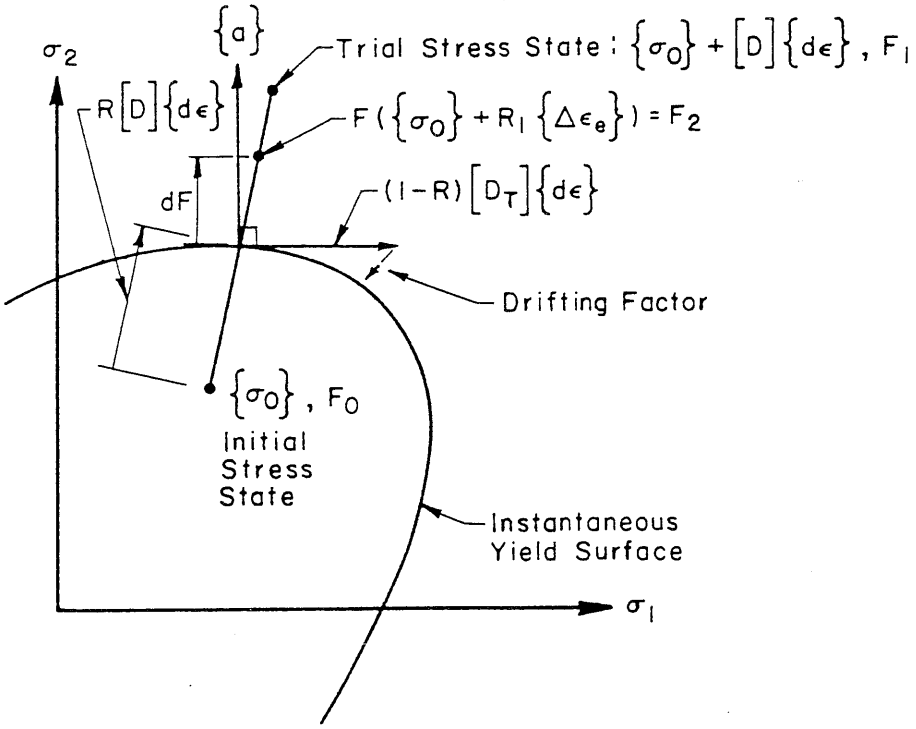


Fig. B.3. Drifting Errors and Transition Between Elastic and Plastic Regions

```
C               POL TABLE DATA TO DEFINE
C
C             MATERIAL MODEL DRUCKER-PRAGER
C
C       GENERAL DRUCKER-PRAGER YIELD FUNCTION FOR PERFECTLY
C       PLASTIC COHESIVE MATERIALS IN 2-D, AXISYMMETRIC, AND 3-D
C
C
C
  DEFINE MATERIAL MODEL DRUCKER_PRAGER
       SUBROUTINE 2
       HISTORY 2 WORDS
       ELEMENTS
             CSTRIANGLE, L2DISOP, Q2DISOP, MQ2DISOP,
             AXIL2DISOP, AXIQ2DISOP, AXIC2DISOP, C2DISOP,
             L3DISOP, Q3DISOP, C3DISOP, MQ3DISOP, Q3DLAGRANGE
       PROPERTIES
              PSTRAIN LOGICAL FALSE
              DEBUGPARMS LOGICAL FALSE
              TRACE LOGICAL FALSE
              TOLERANCE REAL 0.001
              ALPHA      REAL 0.05
              MAXINCR    INTEGER 30
              PRINT     LOGICAL FALSE
              E  REAL 0.0
              NU REAL 0.0
              PHEE REAL 0.0
              COHESION REAL 0.0
              AXISYMMETRIC  LOGICAL FALSE
              DIVERGENCE    INTEGER 500
              COEFFICIENTS  INTEGER  3
       END OF PROPS
       SYMMETRY
     END OF MODEL
```

```
C               POL TABLE DATA TO DEFINE
C
C             MATERIAL MODEL VON MISES
C
C       GENERAL VON MISES MODEL WITH ISOTROPIC HARDENING
C       FOR 2-D, AXISYMMETRIC AND 3-D ELEMENTS
C
C
C
  DEFINE MATERIAL MODEL VON_MISES
       SUBROUTINE 1
       MATERIAL STRESS_STRAIN FUNCTIONS
          SEGMENTAL
       HISTORY 5 WORDS
       ELEMENTS
             CSTRIANGLE, L2DISOP, Q2DISOP, MQ2DISOP,
             AXIL2DISOP, AXIQ2DISOP, AXIC2DISOP, C2DISOP,
             L3DISOP, Q3DISOP, C3DISOP, MQ3DISOP, Q3DLAGRANGE
       PROPERTIES
              PSTRAIN LOGICAL FALSE
              DEBUGPARMS LOGICAL FALSE
              TRACE LOGICAL FALSE
              TOLERANCE REAL 0.001
              ALPHA      REAL 0.05
              MAXINCR    INTEGER 30
              PRINT     LOGICAL FALSE
              AXISYMMETRIC LOGICAL FALSE
              DIVERGENCE INTEGER 500
       END OF PROPS
       SYMMETRY
     END OF MODEL
```

Fig. B.4. Example Material Model Definition Tables for FINITE

```
C                POL TABLE DATA TO DEFINE
C
C             STRESS-STRAIN FUNCTION SEGMENTAL
C
C   ELASTO_PLASTIC OR STRAIGHT LINE SEGEMENT APPROXIMATION
C
C
C
 DEFINE MATERIAL STRESS_STRAIN FUNCTION SEGMENTAL
   SUBROUTINE 1
   PROPERTIES
            YMODULUS  REAL 0.0
            NU REAL 0.0
            SEGMENTS LOGICAL FALSE
            NUMPOINTS INTEGER 0
            STRAINS VECTOR REAL 0.
            STRESSES VECTOR REAL 0.
            ELASTO_PLASTIC  LOGICAL FALSE
            COMPYIELD  REAL 0.0
            TENSYIELD  REAL 0.0
            DEBUG LOGICAL FALSE
    END OF PROPERTIES
  END OF FUNCTION
```

Fig. B.5. Example Stress-Strain Function Definition Tables for FINITE

```
C ***********************************************************************
C *                                                                     *
C *        MATERIAL MODEL # 1 -- VON MISES                              *
C *                                                                     *
C ***********************************************************************
C
C
      SUBROUTINE MTM01( PROPPT,PROPS,HISTRY,DMTRIX,NSTRN,OLDEPS,
     1                  OLIEPS,DEPS,DIEPS,OLDSIG,NEWSIG,
     2                  NWIEPS,GENDAT, TRANS  )
C
C             DECLARE SUBROUTINE PARAMETERS
C
      DIMENSION PROPPT(1), PROPS(1), HISTRY(1), DMTRIX(NSTRN,NSTRN),
     1          OLDEPS(1), OLIEPS(1), DEPS(1), DIEPS(1), OLDSIG(1),
     2          NEWSIG(1), NWIEPS(1), GENDAT(1), TRANS(1)
      REAL NEWSIG, NWIEPS
      INTEGER PROPPT, GENDAT
C
C             DECLARE LOCAL VARIABLES
C
      COMMON /ELECOM/
     1  RWORD, TWOD, AXISYM, THREED, IOUT, REQTYP, ITERNO, SSFNO,
     2  ISFNO, STIME, FTIME, PSTRAN, DEBUG, NPROP, COMVEC(20),
     3  E,     NU,   YIELD, DMEPS(6), IDEPS, EQPEPS, STATE, STYPE,
     4  DELAST(6,6),  DTAU(6),  F0, F1, F2, F3, R, R1, A(6), TAU(6),
     5  SUM, NSEG, DEPSP(6), TRACE,  TOL,  ALPHA,  MAXINC, D(6),
     6  ET, HPRIME, DLAMDA, C,    NSTATE,  UNLOAD, TRAD(6),
     7  ATRAD(6,6), PRINT,  EQTEPS, DTAUEL(6), DIVERG
C
      LOGICAL  THREED, TWOD, AXISYM, PSTRAN, DEBUG, IDEPS,
     1         LWORD, TRACE, UNLOAD, PRINT
C
      INTEGER  REQTYP, SSFNO, ISFNO, STYPE, STATE, DIVERG
C
      REAL     NU, MTMDOT, MTM01F
C
      DIMENSION RCMVEC(1)
C
      EQUIVALENCE  ( RCMVEC,COMVEC ),  ( RWORD,IWORD,LWORD )
C
C
C           ************************************************
C           *                                              *
C           *        INITIALIZE LOCAL VARIABLES            *
C           *                                              *
C           ************************************************
C
C
C
      IOUT = 0
      REQTYP = GENDAT(1)
      ITERNO = GENDAT(2)
      SSFNO  = GENDAT(3)
      ISFNO  = GENDAT(4)
      IWORD  = GENDAT(5)
      STIME  = RWORD
      IWORD  = GENDAT(6)
      FTIME  = RWORD
      GENDAT(7)  = 0
      GENDAT(8)  = 0
      GENDAT(9)  = 0
      IDEPS = GENDAT(11) .GT. 0
C
C             PULL OUT MODEL PROPERTIES
C
      RWORD   = PROPS(PROPPT(1))
      PSTRAN  = LWORD
      RWORD   = PROPS(PROPPT(2))
      DEBUG   = LWORD
      RWORD   = PROPS(PROPPT(3))
      TRACE   = LWORD
      TOL     = PROPS(PROPPT(4))
      ALPHA   = PROPS(PROPPT(5))
      RWORD   = PROPS(PROPPT(6))
      MAXINC  = IWORD
      RWORD   = PROPS(PROPPT(7))
      PRINT   = LWORD
      RWORD   = PROPS(PROPPT(8))
      AXISYM  = LWORD
      RWORD   = PROPS(PROPPT(9))
      DIVERG  = IWORD
      THREED  = NSTRN .EQ. 6
      TWOD = NSTRN .EQ. 3  .OR.  NSTRN .EQ. 4
      IF ( AXISYM ) TWOD = .FALSE.
      IF ( TWOD ) STYPE = 1
      IF ( PSTRAN ) STYPE = 2
      IF ( AXISYM ) STYPE = 3
      IF ( THREED ) STYPE = 4
      IF ( DEBUG .OR. TRACE .OR. PRINT ) CALL MTMERR(IOUT)
      IF (  .NOT. DEBUG )  GO TO 20
      WRITE(IOUT,9000) ( I,GENDAT(I),I=1,10 ), ( I,PROPPT(I),
     1                   I = 1,5 )
      NPROP = 0
      DO 10 I = 1, 5
      IF ( PROPPT(I).GT.NPROP ) NPROP = PROPPT(I)
   10 CONTINUE
      WRITE(IOUT,9002) ( I,PROPS(I),I = 1, NPROP )
C
C             PULL DATA FROM HISTORY VECTOR
```

Fig. B.6. Material Model Von-Mises Subprogram

```
C
   20 CONTINUE
      YIELD  = HISTRY(1)
      EQPEPS = HISTRY(2)
      RWORD  = HISTRY(3)
      STATE  = IWORD
      E      = HISTRY(4)
      NU     = HISTRY(5)
      IF ( .NOT. DEBUG ) GO TO 100
      WRITE(IOUT,9012) YIELD, EQPEPS, STATE, E
C
C               CHECK THE DATA GIVEN BY THE USER. A STRESS/STRAIN
C               FUNCTION MUST BE GIVEN AND NO INITIAL STRAIN
C               FUNCTIONS ARE PERMITTED YET.
C
  100 IF ( SSFNO.GT.0 ) GO TO 110
      IF ( IOUT.EQ.0  ) CALL MTMERR( IOUT )
      WRITE(IOUT,9004)
  110 CONTINUE
C
C
C               BRANCH ON THE TYPE OF REQUEST
C
C
      GO TO ( 200, 800, 400 ), REQTYP
C
C
C
C             ************************************************
C             *                                              *
C             *  INITIALIZE MATERIAL MODEL DATA FOR ELASTIC  *
C             *  STATE AND COMPUTE ELASTIC D MATRIX          *
C             *                                              *
C             ************************************************
C
C
C               CALL THE STRESS/STRAIN FUNCTION TO GET
C               E, NU, AND THE INITIAL YIELD STRESS.
C
C
  200 CONTINUE
      COMVEC(1) = 1
      CALL MTMDSS( COMVEC )
      IF ( COMVEC(2).EQ.0 ) GO TO 210
      IF ( IOUT.EQ.0 ) CALL MTMERR( IOUT )
      WRITE(IOUT,9010)
      GENDAT(9)  = 2
      RETURN
  210 E          = RCMVEC(4)
      NU         = RCMVEC(5)
      YIELD      = RCMVEC(6)
      HISTRY(1) = YIELD
      HISTRY(2) = 0.0
      IWORD     = -1
      HISTRY(3) = RWORD
      HISTRY(4) = E
      HISTRY(5) = NU
      IF ( THREED ) CALL MTM3DD( DMTRIX,E,NU, 6 )
      IF ( TWOD )   CALL MTM2DD( DMTRIX,E,NU,PSTRAN, 4 )
      IF ( AXISYM ) CALL MTMASD( DMTRIX,E,NU, 4 )
      IF ( .NOT. TRACE ) RETURN
      WRITE(IOUT,9006) E, NU, YIELD
      DO 220 I = 1, NSTRN
      WRITE(IOUT,9008) ( DMTRIX(I,J), J = 1, NSTRN )
  220 CONTINUE
      RETURN
C
C
C             ************************************************
C             *                                              *
C             * UPDATE TOTAL STRESSES AT THE STRAIN POINT    *
C             *                                              *
C             ************************************************
C
C
  400 CONTINUE
C
C
C               GET THE ELASTIC D MATRIX. COMPUTE THE INCREMENT
C               OF MECHANICAL STRAIN.  COMPUTE TRIAL STRESSES
C               BASED ON THE MATERIAL PROPERTIES INCORPORATED
C               IN THE CURRENT STRUCTURAL STIFFNESS MATRIX.
C               INITIALIZE NEW STRESSES TO OLD STRESSES
C               FOR PLASTICITY COMPUTATIONS.
C
C
      IF ( TWOD ) CALL MTM2DD( DELAST, E, NU, PSTRAN, 6 )
      IF ( AXISYM ) CALL MTMASD( DELAST, E, NU, 6 )
      IF ( THREED ) CALL MTM3DD( DELAST, E, NU, 6 )
C
      DO 410 I = 1, NSTRN
      DMEPS(I) = DEPS(I)
      NEWSIG(I) = OLDSIG(I)
      IF ( IDEPS ) DMEPS(I) = DMEPS(I) - DIEPS(I)
  410 CONTINUE
C
      CALL MTMVMP( DTAU, DMTRIX, DMEPS, NSTRN, NSTRN )
C
      DO 420 I = 1, NSTRN
      TAU(I) = OLDSIG(I) + DTAU(I)
  420 CONTINUE
```

Fig. B.6. continued

```
      IF ( TRACE ) WRITE(IOUT,9014) ( DMEPS(I), DTAU(I), TAU(I),
     1                               I = 1, NSTRN )
C
C               EVALUATE YIELD CRITERION FOR THE TRIAL
C               STRESSES AND CURRENT YIELD POINT OF UNIAXIAL
C               STRESS/STRAIN CURVE.  IF PREVIOUS STATE
C               OF MATERIAL WAS ELASTIC AND STRESSES
C               ARE INSIDE OR ON THE YIELD SURFACE WITHIN
C               A TOLERANCE, THE CURRENT STATE IS ALSO
C               ELASTIC.  IF PREVIOUS STATE WAS PLASTIC,
C               THE NEW STATE IS ALSO PLASTIC FOR NOW.
C
      F1 = MTM01F( TAU, YIELD, STYPE )
      IF ( STATE .EQ. 1 ) GO TO 480
      IF ( F1 .GT. TOL * YIELD ) GO TO 440
C
C               TRIAL ELASTIC STRESSES ARE INSIDE OR ON THE YIELD SURFACE
C               WITHIN THE TOLERANCE. THE NEW TOTAL STRESSES ARE
C               THE TRIAL STRESSES. UPDATE THE HISTORY AND SET THE
C               FLAGS FOR FINITE.
C
      DO 430 I = 1, NSTRN
      NEWSIG(I) = TAU(I)
  430 CONTINUE
      NEWSIG(NSTRN+1) = F1
      GENDAT(7) = 0
      UNLOAD = .FALSE.
      IF ( TRACE ) WRITE(IOUT,9016) F1, STATE, UNLOAD, ( NEWSIG(I),
     1                                 I = 1, NSTRN )
      IF ( .NOT. PRINT ) RETURN
      WRITE(IOUT,9900)
      WRITE(IOUT,9902)
      WRITE(IOUT,9904)
      WRITE(IOUT,9906) ( NEWSIG(I), I = 1,NSTRN )
      WRITE(IOUT,9908) UNLOAD
      RETURN
C
C               TRIAL ELASTIC STRESSES ARE OUTSIDE THE YIELD SURFACE.
C               COMPUTE THE PORTION OF THE ELASTIC STRESS
C               INCREMENT REQUIRED TO REACH THE YIELD
C               SURFACE.  USE THE DOUBLE R ESTIMATION
C               PROCEDURE OF NAYAK VALID FOR ANY YIELD
C               SURFACE.
C
  440 CONTINUE
C
      F0 = MTM01F( OLDSIG, YIELD, STYPE )
      R  = 0.0
      DO 445 I = 1, NSTRN
      TAU(I)    = OLDSIG(I)
      NEWSIG(I)= OLDSIG(I)
  445 CONTINUE
      IF ( ABS( F0 ) .LE. TOL * YIELD ) GO TO 480
      R1        = -F0 / ( F1 - F0 )
      DO 450 I = 1, NSTRN
      TAU(I)    = OLDSIG(I) + R1 * DTAU(I)
  450 CONTINUE
      F2        = MTM01F( TAU, YIELD, STYPE )
      CALL MTM01A( A, TAU, STYPE )
      R         = R1 - F2 / MTMDOT( A, DTAU, NSTRN )
      DO 470 I = 1, NSTRN
      TAU(I)    = OLDSIG(I) + R * DTAU(I)
      NEWSIG(I) = TAU(I)
  470 CONTINUE
      F3         = MTM01F( TAU, YIELD, STYPE )
      IF ( TRACE ) WRITE(IOUT,9018) F0, F1, R1, F2, R, F3,
     1                  ( A(I), NEWSIG(I), I = 1, NSTRN )
C
C               AFTER COMPLETING THE ABOVE COMPUTATIONS
C               NEWSIG SHOULD LIE VERY CLOSE TO
C               THE YIELD SURFACE. (I.E. F3 = 0 )
C
C
C               COMPUTE ELASTO-PLASTIC STRESSES CORRESPONDING
C               TO THE NON-ELASTIC PART OF THE STRAIN INCREMENT.
C               IF PREVIOUS STATE WAS PLASTIC THIS MEANS
C               THE ENTIRE STRAIN INCREMENT.  USE THE
C               SUBINCREMENT METHOD PROPOSED BY NAYAK.
C               COMPUTE THE NUMBER OF SUBINCREMENTS USING
C               A MAXIMUM ALLOWABLE DEVIATION FROM THE
C               YIELD SURFACE FOR EACH INCREMENT.
C               USER MAY SUPPLY THE FACTOR.
C               DURING SUBINCREMENT METHOD, THE MATERIAL
C               IS HELD CONSTANT AT THE VALUE USED IN THE
C               CURRENT STRUCTURE STIFFNESS MATRIX.
C
  480 CONTINUE
      IF ( STATE .EQ. 1 ) R = 0.0
      IF ( ABS(F1) / ( ALPHA * YIELD )  .LE. DIVERG ) GO TO 485
      CALL MTMERR( IOUT )
      WRITE(IOUT,9928) DIVERG
      GENDAT(9) = 2
      RETURN
  485 NSEG      = IFIX( F1 / ( ALPHA * YIELD ) ) + 1
      IF ( NSEG .LE. 0 ) NSEG = 1
      IF ( NSEG .GT. MAXINC ) NSEG = MAXINC
  490 CONTINUE
      DO 500 I = 1, NSTRN
      DEPSP(I) = ( 1. - R ) * DMEPS(I) / FLOAT( NSEG )
  500 CONTINUE
```

Fig. B.6. continued

```
      CALL MTMVMP( DTAUEL, DELAST, DEPSP, 6, NSTRN )
      IF (TRACE) WRITE(IOUT,9020) ( DEPSP(I), DTAUEL(I), I = 1, NSTRN )
C
C               LOOP FOR EACH SUBINCREMENT, UPDATING STRESSES
C               FOR THE SUBINCREMENT STRAIN INCREMENT.
C
      DO 700 ISEG = 1, NSEG
C
C               INVOKE THE STRESS/STRAIN FUNCTION TO GET THE
C               SLOPE OF THE UNIAXIAL STRESS/STRAIN CURVE FOR
C               THE CURRENT TOTAL STRAIN STRAIN.
C
      COMVEC(1) = 2
      EQTEPS = YIELD/E + EQPEPS
      IF ( EQPEPS .EQ. 0.0 ) EQTEPS = YIELD * ( 1.+TOL )/E
      RCMVEC(3) = EQTEPS
      CALL MTMDSS( COMVEC )
      IF ( COMVEC(2) .EQ. 0 ) GO TO 610
      IF ( IOUT .EQ. 0 ) CALL MTMERR( IOUT )
      WRITE(IOUT,9010)
      GENDAT(9) = 2
      RETURN
C
C               COMPUTE SLOPE OF UNIAXIAL STRESS - PLASTIC STRAIN
C               CURVE
C
  610 ET        = RCMVEC(4)
      HPRIME    = E * ET / ( E - ET )
C
C               GET THE VECTOR NORMAL TO THE YIELD SURFACE - THE
C               CONSTANT BETA, AND COMPUTE DLAMDA, THE EQUIVALENT
C               UNIAXIAL PLASTIC STRAIN INCREMENT.  CHECK IF
C               ELASTIC UNLOADING IS OCCURING ( NEGATIVE PLASTIC
C               STRAIN INCREMENT ).
C
      CALL MTM01A( A, NEWSIG, STYPE )
      CALL MTM01B( A, DELAST, NSTRN, 6, BETA )
      DLAMDA    = MTMDOT( A, DTAUEL, NSTRN ) / ( HPRIME + BETA )
      IF ( TRACE ) WRITE(IOUT,9022) ISEG, ET, HPRIME, DLAMDA,
     1                              BETA, ( A(I),I= 1,NSTRN )
      IF ( DLAMDA .LT. 0.0 ) GO TO 750
C
C               COMPUTE PLASTIC STRAIN INCREMENTS AND ADJUST
C               ELASTIC STRESS INCREMENT TO REFLECT THEM.
C
      CALL MTMVMP( D, DELAST, A, 6, NSTRN )
      DO 620 I = 1, NSTRN
      NEWSIG(I) = NEWSIG(I) + DTAUEL(I) - DLAMDA * D(I)
  620 CONTINUE
C
C
C               UPDATE ACCUMULATED UNIAXIAL PLASTIC STRAIN.
C               IF MATERIAL IS STRAIN HARDENING COMPUTE NEW
C               UNIAXIAL YIELD STRESS.  IF PERFECTLY PLASTIC
C               ADJUST STRESSES SO THEY ARE ON YIELD SURFACE
C               BEFORE STARTING NEW SUBINCREMENT.
C
      EQPEPS = EQPEPS + DLAMDA
      IF ( HPRIME .EQ. 0.0 ) GO TO 630
      YIELD = MTM01F( NEWSIG, 0.0, STYPE )
      GO TO 650
C
  630 F1 = MTM01F( NEWSIG, YIELD, STYPE )
      CALL MTM01A( A, NEWSIG, STYPE )
      C = F1 / MTMDOT( A, A, NSTRN )
      DO 640 I = 1, NSTRN
      NEWSIG(I) = NEWSIG(I) - A(I) * C
  640 CONTINUE
C
  650 IF ( TRACE ) WRITE(IOUT,9024) YIELD, EQPEPS, F1, C,
     1                              ( NEWSIG(I), I = 1, NSTRN )
C
C
  700 CONTINUE
C
C               UPDATE HISTORY FOR STRAIN POINT TO REFLECT THAT
C               MATERIAL IS PLASTIC.  COMPUTE VALUE OF YIELD
C               FUNCTION AND PUT IN STRESS VECTOR.
C
      HISTRY(1) = YIELD
      HISTRY(2) = EQPEPS
      IWORD = 1
      HISTRY(3) = RWORD
      NEWSIG(NSTRN+1) = MTM01F( NEWSIG, YIELD, STYPE )
      GENDAT(7) = 1
      IF ( .NOT. PRINT ) RETURN
      WRITE(IOUT,9900)
      WRITE(IOUT,9910)
      WRITE(IOUT,9904)
      WRITE(IOUT,9906) ( NEWSIG(I), I = 1, NSTRN )
      WRITE(IOUT,9912) YIELD
      WRITE(IOUT,9914) EQPEPS
      F1 = MTM01F( NEWSIG, YIELD, STYPE )
      WRITE(IOUT,9916) F1
      RETURN
C
C               MATERIAL IS UNLOADING ELASTICALLY FROM A PLASTIC
C               STATE.
C
```

Fig. B.6. continued

```
 750  CALL MTMVMP( DTAUEL, DELAST, DMEPS, 6, NSTRN )
      DO 760 I = 1, NSTRN
      NEWSIG(I) = OLDSIG(I) + DTAUEL(I)
 760  CONTINUE
      NEWSIG(NSTRN+1) = MTM01F( NEWSIG, YIELD, STYPE )
      UNLOAD = .TRUE.
      GENDAT(8) = 1
      GENDAT(7) = 0
      IWORD = -1
      HISTRY(3) = BWORD
      IF ( TRACE ) WRITE(IOUT,9914) 0 0,  1, UNLOAD, ( NEWSIG(I),
     1                              I = 1, NSTRN )
      IF ( .NOT. PRINT ) RETURN
      WRITE(IOUT,9900)
      WRITE(IOUT,9902)
      WRITE(IOUT,9904)
      WRITE(IOUT,9906) ( NEWSIG(I), I = 1, NSTRN )
      WRITE(IOUT,9908) UNLOAD
      RETURN
C
C
C           ************************************************
C           *                                              *
C           *        COMPUTE CURRENT D MATRIX              *
C           *                                              *
C           ************************************************
C
C
 800  CONTINUE
C
      IF ( PRINT .OR. TRACE ) WRITE(IOUT,9918)
      IF ( TWOD ) CALL MTM2DD( DMTRIX, E, NU, PSTRAN, 4 )
      IF ( AXISYM ) CALL MTMASD( DMTRIX, E, NU, 4 )
      IF ( THREED ) CALL MTM3DD( DMTRIX, E, NU, 6 )
      IF ( STATE .EQ. -1 ) RETURN
C
C             STRESS AT POINT ARE ON YIELD SURFACE.
C             COMPUTE THE INCREMENTAL STRESS-STRAIN RELATION.
C
C             INVOKE STRESS-STRAIN FUNCTION TO COMPUTE
C             SLOPE OF UNIAXIAL STRESS-STRAIN CURVE FOR CURRENT
C             TOTAL STRAIN.
C
      COMVEC(1) = 2
      EQTEPS = YIELD / E + EQPEPS
      IF ( EQPEPS .EQ. 0.0 ) EQTEPS = YIELD * ( 1.+TOL )/E
      RCMVEC(3) = EQTEPS
      CALL MTMDSS( COMVEC )
      IF ( COMVEC(2) .EQ. 0 ) GO TO 810
      IF ( IOUT .EQ. 0 ) CALL MTMERR( IOUT )
      WRITE(IOUT,9010)
      GENDAT(2) = 2
      RETURN
C
C             COMPUTE SLOPE OF UNIAXIAL STRESS - PLASTIC
C             STRAIN CURVE.
C
 810  ET = RCMVEC(4)
      HPRIME = E * ET / ( E - ET )
C
      DO 820 I = 1, NSTRN
      DO 820 J = 1, NSTRN
      DELAST(I,J) = DMTRIX(I,J)
 820  CONTINUE
      IF ( TRACE ) WRITE(IOUT,9920) ET, HPRIME
      IF ( TRACE ) WRITE(IOUT,9922) ( NEWSIG(I), I = 1, NSTRN )
C
C             GET STRESS VECTOR NORMAL TO YIELD SURFACE FOR
C             CURRENT STATE OF STRESS AND COMPUTE THE BETA
C             CONSTANT.
C
      CALL MTM01A( A, NEWSIG, STYPE )
      CALL MTM01B( A, DELAST, NSTRN, 6, BETA )
      IF ( TRACE ) WRITE(IOUT,9924) BETA
C
C             COMPUTE THE PRODUCT DELAST * A * TRANS(A) * DELAST
C
      DO 830 I = 1,  NSTRN
      TRAD(I) = 0.0
      DO 830 J = 1, NSTRN
      TRAD(I) = TRAD(I) + A(J) * DELAST(J,I)
 830  CONTINUE
      DO 840 I = 1, NSTRN
      DO 840 J = 1, NSTRN
      ATRAD(I,J) =  A(I) * TRAD(J)
 840  CONTINUE
      DO 860 I = 1, NSTRN
      DO 860 J = 1, NSTRN
      SUM = 0.0
      DO 850 L = 1, NSTRN
      SUM = SUM + DELAST(I,L) * ATRAD(L,J)
 850  CONTINUE
      DMTRIX(I,J) = DELAST(I,J) - SUM / ( HPRIME + BETA )
 860  CONTINUE
      IF ( .NOT. ( TRACE .OR. PRINT ) ) RETURN
      WRITE(IOUT,9926)
      DO 870 I = 1, NSTRN
      WRITE(IOUT,9008) ( DMTRIX(I,J), J = 1, NSTRN )
 870  CONTINUE
      RETURN
```

Fig. B.6. continued

```
C
C
 9000 FORMAT(/,5H>>>>>,38H  DEBUG FOR MATERIAL MODEL VON MISES
     1     //,10X,13HGENERAL DATA  /,10(/12X,I3,3X,Z12),/,
     2        10X,17HPROPERTY POINTERS  /,5(/12X,I3,3X,I5) // )
 9002 FORMAT(//,10X,17HPROPERTIES       /,100(/12X,I3,3X,Z12)//)
 9004 FORMAT(10X,42H  A STRESS-STRAIN FUNCTION MUST BE GIVEN.
     1  // )
 9006 FORMAT(   10X,17HE, NU, YIELD                /,3F20.6//)
 9008 FORMAT(   12X,6F15.3 )
 9010 FORMAT(//,40H  MODEL COMPUTATIONS STOPPED BECAUSE OF
     1        /,40H  STRESS-STRAIN FUNCTION ERROR(S).
     2       // )
 9012 FORMAT(//,10X,5HYIELD, 7H EQPEPS, 6H STATE, 3H E , /,
     1          F10.1, F10.6, F3.0, F15.1 )
 9014 FORMAT(//,3X,19H*** UPDATE STRESSES    ,//,
     1 10X, 5HDMEPS, 5H DTAU, 4H TAU, //,
     2 6(/,10X, F10.6, F10.3, F10.3 ) )
 9016 FORMAT(//,5X,16HPOINT IS ELASTIC   ,/,
     1 6X,26HF1, NSTATE, UNLOAD, NEWSIG ,/,
     2 6X, F10.4, I4, L5, 6F10.3 )
 9018 FORMAT(//,5X,32HF0, F1, R1, F2, R, F3, A, NEWSIG   ,/,
     1  6X, 6F10.4, 6(/,6X,F10.6,F10.4) )
 9020 FORMAT(//,5X,13HDEPSP, DTAUEL  /, 6(/,6X,F10.6,F10.3) )
 9022 FORMAT(//,5X,21H***** START INCREMENT    , I5,
     1  /,6X, 27HET, HPRIME, DLAMDA, BETA, A  /, 6X,F10.2,F10.3,
     2  F10.6,F10.2/,3X,6F10.3 )
 9024 FORMAT(//,5X,28HYIELD, EQPEPS, F1, C, NEWSIG                /,
     1  6X,F10.3,F10.6,F10.4,F10.6,/,6X,6F10.3 )
 9900 FORMAT(//,22H0***** MATERIAL STATUS, 1X, 5H***** )
 9902 FORMAT(1H0,5X,16HPOINT IS ELASTIC  )
 9904 FORMAT(1H0,5X,18HNEW TOTAL STRESSES  )
 9906 FORMAT(1H0,7X,6F10.3)
 9908 FORMAT(1H0,7X,15HUNLOADING FLAG  ,L3 )
 9910 FORMAT(1H0,5X,16HPOINT IS PLASTIC  )
 9912 FORMAT(1H0,7X,17HNEW YIELD STRESS  , F10.3 )
 9914 FORMAT(1H0,7X,15HPLASTIC STRAIN  , F10.6 )
 9916 FORMAT(1H0,7X,16HYIELD FUNCTION  , F15.5 )
 9918 FORMAT(//,3X,16H*** NEW D MATRIX       //)
 9920 FORMAT(5X,3HET ,F20.6,/,5X,7HHPRIME , F20.6)
 9922 FORMAT(//,5X,7HNEWSIG ,/,8X,6F10.4 )
 9924 FORMAT(//,5X,5HBETA ,F20.6 )
 9926 FORMAT(//,5X,8HD MATRIX   )
 9928 FORMAT(/,3X,40HNUMBER OF SUBINCREMENTS IS GREATER THAN ,I4,
     1       /,2X,40HSTIFFNESS IS PROBABLY SINGULAR.             //)
C
      END
```

```
C ***********************************************************************
C *                                                                     *
C *       MATERIAL MODEL # 1 ROUTINE -- MTM01F                          *
C *                                                                     *
C ***********************************************************************
C
C
      REAL FUNCTION MTM01F( TAU, YIELD, STYPE )
C
C
C              COMPUTE THE VALUE OF THE HUBER-MISES YIELD
C              FUNCTION FOR THE STATE OF STRESS TAU AND
C              CURRENT UNIAXIAL YIELD STRESS YIELD.
C
C
      REAL  TAU(1), YIELD, S(6), SM
      INTEGER STYPE
C
C              GET THE DEVIATOR STRESSES AND MEAN STRESS.
C
      CALL MTMDEV( TAU, STYPE, S, SM )
C
      ROOT3 = SQRT( 3.0 )
C
      GO TO ( 10, 30, 30, 40 ), STYPE
C
 10   MTM01F = ROOT3 * SQRT( 0.5*( S(1)*S(1) + S(2)*S(2) + SM*SM ) +
     1                       S(3)*S(3) ) - YIELD
      RETURN
C
 30   MTM01F = ROOT3 * SQRT( 0.5*( S(1)*S(1) + S(2)*S(2) + S(4)*S(4) )
     1                       + S(3)*S(3) ) - YIELD
      RETURN
C
 40   MTM01F = ROOT3 * SQRT( 0.5*( S(1)*S(1) + S(2)*S(2) + S(3)*S(3) )
     1                       + S(4)*S(4) + S(5)*S(5) + S(6)*S(6) )
     2                  - YIELD
      RETURN
C
      END
```

Fig. B.6. continued

```
C ***********************************************************************
C *                                                                     *
C *     MATERIAL MODEL # 1 ROUTINE -- MTM01A                            *
C *                                                                     *
C ***********************************************************************
C
C
      SUBROUTINE MTM01A( A, TAU, STYPE )
C
C
C             COMPUTE COMPONENTS OF THE STRESS VECTOR NORMAL
C             TO THE HUBER-MISES YIELD FUNCTION.
C             STRESSES TAU ARE ASSUMED TO LIE ON YIELD SURFACE.
C
C
      REAL A(1), TAU(1), S(6)
      INTEGER STYPE
C
C             GET THE DEVIATORIC STRESSES IN THE S VECTOR.
C
      CALL MTMDEV( TAU, STYPE, S, SM )
C
      ROOT3 = SQRT( 3. )
C
      GO TO ( 10, 30, 30, 40 ), STYPE
C
 10   SBAR = SQRT( 0.5*( S(1)*S(1) + S(2)*S(2) + SM*SM ) + S(3)*S(3) )
      A(1) = ROOT3 * S(1) / ( 2. * SBAR )
      A(2) = ROOT3 * S(2) / ( 2. * SBAR )
      A(3) = ROOT3 * S(3) / SBAR
      A(4) = 0.0
      RETURN
C
 30   SBAR = SQRT( 0.5*( S(1)*S(1) + S(2)*S(2) + S(4)*S(4) )
     1             + S(3)*S(3) )
      A(1) = ROOT3 * S(1) / ( 2. * SBAR )
      A(2) = ROOT3 * S(2) / ( 2. * SBAR )
      A(3) = ROOT3 * S(3) / SBAR
      A(4) = ROOT3 * S(4) / ( 2. * SBAR )
      RETURN
C
 40   SBAR = SQRT( 0.5*( S(1)*S(1) + S(2)*S(2) + S(3)*S(3) ) +
     1              S(4)*S(4) + S(5)*S(5) + S(6)*S(6) )
      DO 45 I = 1, 6
      D = 1.0
      IF ( I.GT. 3 ) D = 2.0
      A(I) = ROOT3 * D * S(I) / ( 2. * SBAR )
 45   CONTINUE
      RETURN
C
      END
```

```
C ***********************************************************************
C *                                                                     *
C *      MATERIAL MODEL # 1 ROUTINE -- MTM01B                           *
C *                                                                     *
C ***********************************************************************
C
C
      SUBROUTINE MTM01B( A, D, NROW, NROWD, BETA )
C
C
C             COMPUTE THE PRODUCT TRANS( A ) * D * A
C             A IS STRESS VECTOR NORMAL TO YIELD SURFACE.
C             D IS ELASTIC STRESS-STRAIN MATRIX.
C
C
      REAL  A(1), D(NROWD,1), BETA
C
C
      BETA = 0.0
      DO 20 I = 1, NROW
      SUM = 0.0
      DO 10 J = 1, NROW
      SUM = SUM + D(I,J) * A(J)
 10   CONTINUE
      BETA = BETA + A(I) * SUM
 20   CONTINUE
C
      RETURN
      END
```

Fig. B.6. continued

```
      SUBROUTINE MTS01( COMVEC, PROPPT, PROPS, GENDAT )
      DIMENSION COMVEC(1), PROPPT(1), PROPS(1), GENDAT(1)
      REAL PROPS
      INTEGER  PROPPT, GENDAT
      EQUIVALENCE ( A,LA,IA )
      LOGICAL LA
C
      COMMON /FUNCOM/ A, REQTYP, SETUP, NEWD, SLOPE, EPSBAR, E,
     1                NU, SEGS, NUMPTS, EPSLOC, SIGLOC, ELSPLS,
     2                COMYLD, TENYLD, DEBUG, IOUT, NPROP, I,
     3                SIGMA, ET, YLD
      REAL COMVEC, NU
      INTEGER  REQTYP, EPSLOC, SIGLOC
      LOGICAL  SETUP, NEWD, SLOPE, SEGS, ELSPLS, DEBUG
C
C
C             STRESS-STRAIN FUNCTION # 1 - UNIAXIAL
C
C             FOR USE WITH MATERIAL MODELS REQUIRING
C             DATA FROM THE  UNIAXIAL STRESS-STRAIN
C             CURVE OF THE MATERIAL.
C
C
C             LOOK AT REQUEST FROM THE MATERIAL MODEL
C
      IOUT = 0
      A = COMVEC(1)
      REQTYP = IA
      SETUP = REQTYP.EQ.1
      NEWD  = REQTYP.EQ.2
      SLOPE = REQTYP.EQ.3
      EPSBAR = COMVEC(3)
C
C             PULL OUT SOME PROPERTIES AND SET LOCATIONS
C             OF OTHERS.
C
      E = PROPS(PROPPT(1))
      NU    = PROPS(PROPPT(2))
      A     = PROPS(PROPPT(3))
      SEGS = LA
      A     = PROPS(PROPPT(4))
      NUMPTS = IA
      EPSLOC = PROPPT(5)
      SIGLOC = PROPPT(6)
      A     = PROPS(PROPPT(7))
      ELSPLS = LA
      COMYLD = PROPS(PROPPT(8))
      TENYLD = PROPS(PROPPT(9))
      A      = PROPS(PROPPT(10))
      DEBUG = LA
C
C             IF DEBUGGING WRITE OUT DATA.
C
      IF ( .NOT.DEBUG ) GO TO 20
      CALL MTSSER( IOUT )
      WRITE(IOUT,1000) ( COMVEC(I),I=1,7 ), ( PROPPT(I),I=1,9 )
      NPROP = 0
      DO 10 I = 1, 9
      IF ( PROPPT(I).GT.NPROP ) NPROP = PROPPT(I)
   10 CONTINUE
      WRITE(IOUT,1010) ( I, PROPS(I), I= 1, NPROP )
C
C
   20 IF ( .NOT.SETUP ) GO TO 100
C
C             SET UP.  RETURN INITIAL SLOPE OF STRESS-STRAIN
C             CURVE, INITIAL POISSON'S RATIO AND THE FIRST TENSION
C             YIELD STRESS.
C
      IF ( .NOT.ELSPLS ) GO TO 27
      COMVEC(4) = E
      COMVEC(5) = NU
      COMVEC(6) = TENYLD
      GO TO 26
   27 COMVEC(4) = E
      COMVEC(5) = NU
      DO 30 I = 1, NUMPTS
      IF ( PROPS(SIGLOC+I-1) .LE. 0.0 ) GO TO 30
      COMVEC(6) = PROPS(SIGLOC+I-1)
      GO TO 26
   30 CONTINUE
      IF ( IOUT .EQ. 0 ) CALL MTSSER( IOUT )
      WRITE(IOUT,1040)
      IA = 1
      COMVEC(2) = A
      RETURN
   26 IA = 0
      COMVEC(2) = A
      GO TO 2000
C
C
  100 IF ( .NOT. NEWD ) GO TO 500
C
C
C             GIVEN TOTAL STRAIN, EPSBAR, RETURN TANGENT MODULUS,
C             NEXT YIELD STRESS, POISSON RATIO, AND CORRESPONDING
C             STRESS.
C
C
      IA = 0
```

Fig. B.7. Stress-Strain Function SEGMENTAL Subprogram

```
      COMVEC(2) = A
      IF ( .NOT. ELSPLS ) GO TO 150
      IF ( EPSBAR ) 110, 120, 130
  110 SIGMA = E * EPSBAR
      COMVEC(4) = E
      COMVEC(5) = NU
      COMVEC(6) = COMYLD
      COMVEC(8) = SIGMA
      IF ( ABS(SIGMA) .LE. ABS(COMYLD) ) GO TO 2000
      COMVEC(4) = 0.0
      COMVEC(8) = COMYLD
      GO TO 2000
  120 COMVEC(4) = E
      COMVEC(5) = NU
      COMVEC(6) = TENYLD
      COMVEC(8) = 0.0
      GO TO 2000
  130 COMVEC(4) = E
      COMVEC(5) = NU
      COMVEC(6) = TENYLD
      SIGMA = EPSBAR * E
      COMVEC(8) = SIGMA
      IF ( SIGMA .LE. TENYLD ) GO TO 2000
      COMVEC(4) = 0.0
      COMVEC(8) = TENYLD
      GO TO 2000
C
C
C             SEGMENTAL STRESS-STRAIN CURVE.
C
C
  150 CALL MTS01A( NUMPTS,PROPS(EPSLOC),PROPS(SIGLOC),EPSBAR,
     1             SIGMA,ET,YLD )
      COMVEC(4) = ET
      COMVEC(5) = NU
      COMVEC(6) = YLD
      COMVEC(8) = SIGMA
      GO TO 2000
C
C
C             ONLY  SETUP AND NEWD IMPLEMENTED.
C
C
  500 IA = 0
      COMVEC(2) = A
      GO TO 2000
C
C
C             DONE
C
C
 2000 IF ( .NOT. DEBUG ) RETURN
      WRITE(IOUT,1020) ( COMVEC(I),I=1,8 )
      RETURN
      END
```

```
      SUBROUTINE MTS01A( NUMPTS, EPS, SIGMA, EPSBAR, SIGBAR, ET,
     1                   YLD )
      DIMENSION  EPS(1), SIGMA(1)
C
C
C             COMPUTE THE UNIAXIAL STRESS, TANGENT MODULUS,
C             AND NEXT YIELD POINT FOR EPSBAR FOR A SEGMENTAL
C             UNIAXIAL STRESS-STRAIN CURVE.
C
C
C
C             CHECK LIMITS.
C
C
      IF ( EPSBAR .GT. EPS(1) ) GO TO 10
      ET = 0.0
      SIGBAR = SIGMA(1)
      YLD = SIGBAR
      RETURN
C
   10 IF ( EPSBAR .LT. EPS(NUMPTS) ) GO TO 20
      ET = 0.0
      SIGBAR = SIGMA(NUMPTS)
      YLD = SIGBAR
      RETURN
C
   20 DO 30 I = 1, NUMPTS
      IF ( EPSBAR .LT. EPS(I) ) GO TO 40
   30 CONTINUE
   40 DEPS = EPS(I) - EPS(I-1)
      DSIG = SIGMA(I) - SIGMA(I-1)
      ET = DSIG / DEPS
      SIGBAR = SIGMA(I-1) + ET * ( EPSBAR-EPS(I-1) )
      IF ( SIGBAR ) 41, 42, 42
   41 YLD = SIGMA(I-1)
      RETURN
   42 YLD = SIGMA(I)
      RETURN
C
      END
```

Fig. B.7. continued

ONR DISTRIBUTION LIST

PART 1 - Government

Administrative and Liaison Activities

Office of Naval Research
Department of the Navy
Arlington, VA 22217
Attn: Code 474 (2)
Code 471
Code 200

Director
Office of Naval Research
Branch Office
666 Summer Street
Boston, MA 02210

Director
Office of Naval Research
Branch Office
536 South Clark Street
Chicago, IL 60605

Director
Office of Naval Research
New York Area Office
715 Broadway - 5th Floor
New York, NY 10003

Director
Office of Naval Research
Branch Office
1030 East Green Street
Pasadena, CA 91106

Naval Research Laboratory (6)
Code 2627
Washington, DC 20375

Defense Documentation Center (12)
Cameron Station
Alexandria, VA 22314

Navy

Undersea Explosion Research Division
Naval Ship Research and Development
Center
Norfolk Naval Shipyard
Portsmouth, VA 23709
Attn: Dr. E. Palmer, Code 177

Navy (Con't.)

Naval Research Laboratory
Washington, DC 20375
Attn: Code 8400
8410
8430
8440
6300
6390
6380

David W. Taylor Naval Ship Research
and Development Center
Annapolis, MD 21402
Attn: Code 2740
28
281

U.S. Naval Weapons Center
China Lake, CA 93555
Attn: Code 4062
4520

Commanding Officer
U.S. Naval Civil Engineering Laboratory
Code L31
Port Hueneme, CA 93041

Naval Surface Weapons Center
White Oak
Silver Spring, MD 20910
Attn: Code R-10
G-402
K-82

Technical Director
Naval Ocean Systems Center
San Diego, CA 92152

Supervisor of Shipbuilding
U.S. Navy
Newport News, VA 23607

U.S. Navy Underwater Sound
Reference Division
Naval Research Laboratory
P.O. Box 8337
Orlando, FL 32806

Navy (Con't.)

Chief of Naval Operations
Department of the Navy
Washington, DC 20350
Attn: Code OP-098

Strategic Systems Project Office
Department of the Navy
Washington, DC 20376
Attn: NSP-200

Naval Air Systems Command
Department of the Navy
Washington, DC 20361
Attn: Code 5302 (Aerospace and Structures)
604 (Technical Library)
320B (Structures)

Naval Air Development Center
Director, Aerospace Mechanics
Warminster, PA 18974

U.S. Naval Academy
Engineering Department
Annapolis, MD 21402

Naval Facilities Engineering Command
200 Stovall Street
Alexandria, VA 22332
Attn: Code 03 (Research and Development)
04B
045
14114 (Technical Library)

Naval Sea Systems Command
Department of the Navy
Washington, DC 20362
Attn: Code 03 (Research and Technology)
037 (Ship Silencing Division)
035 (Mechanics and Materials)

Naval Ship Engineering Center
Department of the Navy
Washington, DC 20362
Attn: Code 6105G
6114
6120D
6128
6129

Commanding Officer and Director
David W. Taylor Naval Ship
Research and Development Center
Bethesda, MD 20084
Attn: Code 042
17
172
173
174
1800
1844
1102.1
1900
1901
1945
1960
1962

Naval Underwater Systems Center
Newport, RI 02840
Attn: Dr. R. Trainor

Naval Surface Weapons Center
Dahlgren Laboratory
Dahlgren, VA 22448
Attn: Code G04
G20

Technical Director
Mare Island Naval Shipyard
Vallejo, CA 94592

U.S. Naval Postgraduate School
Library
Code 0384
Monterey, CA 93940

Webb Institute of Naval Architecture
Attn: Librarian
Crescent Beach Road, Glen Cove
Long Island, NY 11542

Army

Commanding Officer (2)
U.S. Army Research Office
P.O. Box 12211
Research Triangle Park, NC 27709
Attn: Mr. J. J. Murray,
CRD-AA-IP

Watervliet Arsenal
MAGGS Research Center
Watervliet, NY 12189
Attn: Director of Research

U.S. Army Materials and Mechanics
Research Center
Watertown, MA 02172
Attn: Dr. R. Shea, DRXMR-T

U.S. Army Missile Research and
Development Center
Redstone Scientific Information
Center
Chief, Document Section
Redstone Arsenal, AL 35809

Army Research and Development
Center
Fort Belvoir, VA 22060

Air Force

Commander WADD
Wright-Patterson Air Force Base
Dayton, OH 45433
Attn: Code WWRMDD
AFFDL (FDDS)
Structures Division
AFLC (MCEEA)

Chief Applied Mechanics Group
U.S. Air Force Institute of Technology
Wright-Patterson Air Force Base
Dayton, OH 45433

Chief, Civil Engineering Branch
WLRC, Research Division
Air Force Weapons Laboratory
Kirtland Air Force Base
Albuquerque, NM 87117

Air Force Office of Scientific Research
Bolling Air Force Base
Washington, DC 20332
Attn: Mechanics Division

Department of the Air Force
Air University Library
Maxwell Air Force Base
Montgomery, AL 36112

NASA

National Aeronautics and Space Administration
Structures Research Division
Langley Research Center
Langley Station
Hampton, VA 23365

National Aeronautics and Space Administration
Associate Administrator for Advanced
Research and Technology
Washington, DC 20546

Scientific and Technical Information Facility
NASA Representative (S-AK/DL)
P.O. Box 5700
Bethesda, MD 20014

Other Government Activities

Commandant
Chief, Testing and Development Division
U.S. Coast Guard
1300 E Street, NW
Washington, DC 20226

Technical Director
Marine Corps Development
and Education Command
Quantico, VA 22134

Director Defense Research
and Engineering
Technical Library
Room 3C128
The Pentagon
Washington, DC 20301

Director
National Bureau of Standards
Washington, DC 20034
Attn: Mr. B. L. Wilson, EM 219

Dr. M. Gaus
National Science Foundation
Environmental Research Division
Washington, DC 20550

Library of Congress
Science and Technology Division
Washington, DC 20540

Director
Defense Nuclear Agency
Washington, DC 20305
Attn: SPSS

Mr. Jerome Persh
Staff Specialist for Materials
and Structures
OUSDR&E, The Pentagon
Room 3D1089
Washington, DC 20301

Chief, Airframe and Equipment Branch
FS-120
Office of Flight Standards
Federal Aviation Agency
Washington, DC 20553

National Academy of Sciences
National Research Council
Ship Hull Research Committee
2101 Constitution Avenue
Washington, DC 20418
Attn: Mr. A. R. Lytle

National Science Foundation
Engineering Mechanics Section
Division of Engineering
Washington, DC 20550

Picatinny Arsenal
Plastics Technical Evaluation Center
Attn: Technical Information Section
Dover, NJ 07801

Maritime Administration
Office of Maritime Technology
14th and Constitution Ave., NW
Washington, DC 20230

Maritime Administration
Office of Ship Construction
14th and Constitution Ave., NW
Washington, DC 20230

PART 2 - Contractors and Other Technical Collaborators

Universities

Dr. J. Tinsley Oden
University of Texas at Austin
345 Engineering Science Building
Austin, TX 78712

Professor Julius Miklowitz
California Institute of Technology
Division of Engineering
and Applied Sciences
Pasadena, CA 91109

Dr. Harold Liebowitz, Dean
School of Engineering and
Applied Science
George Washington University

Professor Eli Sternberg
California Institute of Technology
Division of Engineering and
Applied Sciences
Pasadena, CA 91109

Professor Paul M. Naghdi
University of California
Department of Mechanical Engineering
Berkeley, CA 94720

Professor A. J. Durelli
Oakland University
School of Engineering
Rochester, MI 48063

Professor F. L. DiMaggio
Columbia University
Department of Civil Engineering
New York, NY 10027

Professor Norman Jones
Massachusetts Institute of Technology
Department of Ocean Engineering
Cambridge, MA 02139

Professor E. J. Skudrzyk
Pennsylvania State University
Applied Research Laboratory
Department of Physics
State College, PA 16801

Professor J. Kempner
Polytechnic Institute of New York
Department of Aerospace Engineering and
Applied Mechanics
333 Jay Street
Brooklyn, NY 11201

Professor J. Klosner
Polytechnic Institute of New York
Department of Aerospace Engineering and
Applied Mechanics
333 Jay Street
Brooklyn, NY 11201

Professor R. A. Schapery
Texas A&M University
Department of Civil Engineering
College Station, TX 77843

Professor Walter D. Pilkey
University of Virginia
Research Laboratories for the
Engineering Sciences
School of Engineering and
Applied Sciences
Charlottesville, VA 22901

Professor K. D. Willmert
Clarkson College of Technology
Department of Mechanical Engineering
Potsdam, NY 13676

Dr. Walter E. Haisler
Texas A&M University
Aerospace Engineering Department
College Station, TX 77843

Dr. Hussein A. Kamel
University of Arizona
Department of Aerospace and
Mechanical Engineering
Tucson, AZ 85721

Dr. S. J. Fenves
Carnegie-Mellon University
Department of Civil Engineering
Schenley Park
Pittsburgh, PA 15213

Universities (Con't.)

Dr. Ronald L. Huston
Department of Engineering Analysis
University of Cincinnati
Cincinnati, OH 45221

Professor G. C. M. Sih
Lehigh University
Institute of Fracture and
Solid Mechanics
Bethlehem, PA 18015

Professor Albert S. Kobayashi
University of Washington
Department of Mechanical Engineering
Seattle, WA 98105

Professor Daniel Frederick
Virginia Polytechnic Institute and
State University
Department of Engineering Mechanics
Blacksburg, VA 24061

Professor A. C. Eringen
Princeton University
Department of Aerospace and
Mechanical Sciences
Princeton, NJ 08540

Professor E. H. Lee
Stanford University
Division of Engineering Mechanics
Stanford, CA 94305

Professor Albert I. King
Wayne State University
Biomechanics Research Center
Detroit, MI 48202

Dr. V. R. Hodgson
Wayne State University
School of Medicine
Detroit, MI 48202

Dean B. A. Boley
Northwestern University
Department of Civil Engineering
Evanston, IL 60201

Professor P. G. Hodge, Jr.
University of Minnesota
Department of Aerospace Engineering
and Mechanics
Minneapolis, MN 55455

Dr. D. C. Drucker
University of Illinois
Dean of Engineering
Urbana, IL 61801

Professor N. M. Newmark
University of Illinois
Department of Civil Engineering
Urbana, IL 61803

Professor E. Reissner
University of California, San Diego
Department of Applied Mechanics
La Jolla, CA 92037

Professor William A. Nash
University of Massachusetts
Department of Mechanics and
Aerospace Engineering
Amherst, MA 01002

Professor G. Herrmann
Stanford University
Department of Applied Mechanics
Stanford, CA 94305

Professor J. D. Achenbach
Northwestern University
Department of Civil Engineering
Evanston, IL 60201

Professor S. B. Dong
University of California
Department of Mechanics
Los Angeles, CA 90024

Professor Burt Paul
University of Pennsylvania
Towne School of Civil and
Mechanical Engineering
Philadelphia, PA 19104

Universities (Con't.)

Professor H. W. Liu
Syracuse University
Department of Chemical Engineering
and Metallurgy
Syracuse, NY 13210

Professor S. Bodner
Technion R&D Foundation
Haifa, Israel

Professor Werner Goldsmith
University of California
Department of Mechanical Engineering
Berkeley, CA 94720

Professor R. S. Rivlin
Lehigh University
Center for the Application
of Mathematics
Bethlehem, PA 18015

Professor F. A. Cozzarelli
State University of New York at Buffalo
Division of Interdisciplinary Studies
Karr Parker Engineering Building
Chemistry Road
Buffalo, NY 14214

Professor Joseph L. Rose
Drexel University
Department of Mechanical Engineering
and Mechanics
Philadelphia, PA 19104

Professor B. K. Donaldson
University of Maryland
Aerospace Engineering Department
College Park, MD 20742

Professor Joseph A. Clark
Catholic University of America
Department of Mechanical Engineering
Washington, DC 20064

Professor T. C. Huang
University of Wisconsin-Madison
Department of Engineering Mechanics
Madison, WI 53706

Dr. Samuel B. Batdorf
University of California
School of Engineering
and Applied Science
Los Angeles, CA 90024

Professor Isaac Fried
Boston University
Department of Mathematics
Boston, MA 02215

Professor Michael Pappas
New Jersey Institute of Technology
Newark College of Engineering
323 High Street
Newark, NJ 07102

Professor E. Krempl
Rensselaer Polytechnic Institute
Division of Engineering
Engineering Mechanics
Troy, NY 12181

Dr. Jack R. Vinson
University of Delaware
Department of Mechanical and Aerospace
Engineering and the Center for
Composite Materials
Newark, DL 19711

Dr. Dennis A. Nagy
Princeton University
School of Engineering and Applied Science
Department of Civil Engineering
Princeton, NJ 08540

Dr. J. Duffy
Brown University
Division of Engineering
Providence, RI 02912

Dr. J. L. Swedlow
Carnegie-Mellon University
Department of Mechanical Engineering
Pittsburgh, PA 15213

Dr. V. K. Varadan
Ohio State University Research Foundation
Department of Engineering Mechanics
Columbus, OH 43210

Universities (Con't.)

Dr. Z. Hashin
University of Pennsylvania
Department of Metallurgy and
Materials Science
College of Engineering and
Applied Science
Philadelphia, PA 19104

Dr. Jackson C. S. Yang
University of Maryland
Department of Mechanical Engineering
College Park, MD 20742

Professor T. Y. Chang
University of Akron
Department of Civil Engineering
Akron, OH 44325

Professor Charles W. Bert
University of Oklahoma
School of Aerospace, Mechanical,
and Nuclear Engineering
Norman, OK 73019

Professor Satya N. Atluri
Georgia Institute of Technology
School of Engineering Science and
Mechanics
Atlanta, GA 30332

Professor Graham F. Carey
University of Texas at Austin
Department of Aerospace Engineering
and Engineering Mechanics
Austin, TX 78712

Industry and Research Institutes

Dr. Jackson C. S. Yang
Advanced Technology and Research, Inc.
10006 Green Forest Drive
Adelphi, MD 20783

Dr. Norman Hobbs
Kaman AviDyne
Division of Kaman
Sciences Corp.
Burlington, MA 01803

Industry and Research Institutes (Con't.)

Argonne National Laboratory
Library Services Department
9700 South Cass Avenue
Argonne, IL 60440

Dr. M. C. Junger
Cambridge Acoustical Associates
1033 Massachusetts Avenue
Cambridge, MA 02138

Dr. V. Godino
General Dynamics Corporation
Electric Boat Division
Groton, CT 06340

Dr. J. E. Greenspon
J. G. Engineering Research Associates
3831 Menlo Drive
Baltimore, MD 21215

Dr. K. C. Park
Lockheed Missile and Space Company
3251 Hanover Street
Palo Alto, CA 94304

Newport News Shipbuilding and
Dry Dock Company
Library
Newport News, VA 23607

Dr. W. F. Bozich
McDonnell Douglas Corporation
5301 Bolsa Avenue
Huntington Beach, CA 92647

Dr. H. N. Abramson
Southwest Research Institute
8500 Culebra Road
San Antonio, TX 78284

Dr. R. C. DeHart
Southwest Research Institute
8500 Culebra Road
San Antonio, TX 78284

Dr. M. L. Baron
Weidlinger Associates
110 East 59th Street
New York, NY 10022

Industry and Research Institutes (Con't.)

Dr. T. L. Geers
Lockheed Missiles and Space Company
3251 Hanover Street
Palo Alto, CA 94304

Mr. William Caywood
Applied Physics Laboratory
Johns Hopkins Road
Laurel, MD 20810

Dr. Robert E. Nickell
Pacifica Technology
P.O. Box 148
Del Mar, CA 92014

Dr. M. F. Kanninen
Battelle Columbus Laboratories
505 King Avenue
Columbus, OH 43201

Dr. G. T. Hahn
Battelle Columbus Laboratories
505 King Avenue
Columbus, OH 43201

Dr. A. A. Hochrein
Daedalean Associates, Inc.
Springlake Research Center
15110 Frederick Road
Woodbine, MD 21797

Mr. Richard Y. Dow
National Academy of Sciences
2101 Constitution Avenue
Washington, DC 20418

Mr. H. L. Kington
Airesearch Manufacturing Company
of Arizona
P.O. Box 5217
111 South 34th Street
Phoenix, AZ 85010

Dr. M. H. Rice
Systems, Science, and Software
P.O. Box 1620
La Jolla, CA 92037

SECURITY CLASSIFICATION OF THIS PAGE *(When Data Entered)*

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER UILU-ENG-78-2020	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE *(and Subtitle)* Numerical and Software Requirements for General Nonlinear Finite Element Analysis		5. TYPE OF REPORT & PERIOD COVERED Technical Report 6. PERFORMING ORG. REPORT NUMBER SRS No. 454
7. AUTHOR(s) R. H. Dodds Jr. L. A. Lopez D. A. Pecknold		8. CONTRACT OR GRANT NUMBER(s) N00014-75-C-0164
9. PERFORMING ORGANIZATION NAME AND ADDRESS Dept. of Civil Engineering University of Illinois Urbana, Illinois 61801		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS Project No. NR-064-183
11. CONTROLLING OFFICE NAME AND ADDRESS Office of Naval Research, Code N00014 Department of the Navy Arlington, Virginia 22217		12. REPORT DATE September 1978 13. NUMBER OF PAGES 195
14. MONITORING AGENCY NAME & ADDRESS *(if different from Controlling Office)* Material Science Division Structural Mechanics Division (Code 474) Office of Naval Research (800 Quincy St.) Arlington, Virginia 22217		15. SECURITY CLASS. *(of this report)* UNCLASSIFIED 15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT *(of this Report)*

Approved for Public Release: Distribution Unlimited

17. DISTRIBUTION STATEMENT *(of the abstract entered in Block 20, if different from Report)*

18. SUPPLEMENTARY NOTES

19. KEY WORDS *(Continue on reverse side if necessary and identify by block number)*

Nonlinear Analysis
Finite Element
Software

20. ABSTRACT *(Continue on reverse side if necessary and identify by block number)*

Until recently only the theoretical aspects of formulating the analysis techniques were considered worthy topics for University research efforts. Proper design and implementation of user-oriented computer software for finite element analysis has now become recognized as an essential component of the analysis process. In this work, the theoretical formulations are again considered, however, primary emphasis has been placed on design and implementation of the computer software.

(cont.)

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE *(When Data Entered)*

20. (cont).

The governing equations of nonlinear continuum mechanics, usually expressed in tensor form, are given in matrix notation familiar to most structural engineers. Both geometric and material nonlinearities are considered.

The classical equations of the Lagrangian description are cast into the finite element method. Specific matrices are given for large deformations of a three dimensional solid to illustrate the procedure.

A general purpose, user-oriented software system, FINITE, for static linear and nonlinear analysis is described. FINITE relies upon the POLO supervisor for problem-oriented-language translation, data management, and dynamic memory allocation. FINITE supports multi-level user-defined substructuring and condensation for linear and nonlinear analysis. Isolation of the element and material model libraries from the system enables rapid entry of new modeling capabilities.

A number of nonlinear example analyses are presented to demonstrate features of the FINITE system.